REPORT DOCUMENTATION PAGE

AFRL-SR-AR-TR-04-

0028

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
30-11-2003	Final Technical Report	12/01/2000 to 11/30/2003

4. TITLE AND SUBTITLE
Designing Optimal Generalized Hill Climbing Algorithms with Applications to Discrete Manufacturing Process Design Optimization

5a. CONTRACT NUMBER

5b. GRANT NUMBER
F49620-01-1-0007

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Sheldon H. Jacobson

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of Illinois
Urbana, IL

8. PERFORMING ORGANIZATION REPORT NUMBER
#1

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Office of Scientific Research
4015 Wilson Blvd., Room 824A
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)
AFOSR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
None

DISTRIBUTION STATEMENT A
Approved for Public Release
Distribution Unlimited

20040130 070

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Generalized hill climbing (GHC) algorithms provide a well-defined framework to model and study the performance of algorithms for intractable discrete optimization problems. For ease of analysis, such algorithms have either been studied empirically (involving extensive trial and error on specific problem instances) or asymptotically (hence leading to convergence results as the number of iterations grows). Unfortunately, such results provide limited insight into the finite-time performance of such algorithms, nor provide guidance for practitioners on how to design and implement such algorithms to optimize their performance given a limited amount of computing time and resources. New algorithms have been developed to address sets of related discrete optimization problems. The primary applications for these algorithms are discrete manufacturing process design optimization problems and construction site-level problems of interest to the Air Force. New performance criteria (both asymptotic and finite-time) have also been developed to analyze these and other related GHC algorithms.

15. SUBJECT TERMS
Discrete Optimization, Heuristics, Earth Moving, Manufacturing Process Design Optimization

16. SECURITY CLASSIFICATION OF: Unclassified			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE	None	45	Sheldon H. Jacobson
Unclassified	Unclassified	Unclassified			19b. TELEPHONE NUMBER (include area code) 217-244-7275

FINAL TECHNICAL REPORT

Submitted to

Juan R. Vasquez, Lt. Colonel, United States Air Force, Ph.D.
Program Manager: Optimization and Discrete Mathematics
Directorate of Mathematics and Space Sciences
Air Force Office of Scientific Research
4015 Wilson Blvd., Room 824A
Arlington, VA 22203-1954
juan.vasquez@afosr.af.mil

Principal Investigator:

Sheldon H. Jacobson, Ph.D.
Department of Mechanical and Industrial Engineering
University of Illinois
Urbana, IL 61801-2906
(217) 244-7275 (office)
(217) 244-6534 (fax)
shj@uiuc.edu
www.staff.uiuc.edu/~shj/shj.html

Grant Number:

F49620-01-1-0007

Table of Contents

EXECUTIVE SUMMARY

The research conducted under this grant focused on the design and analysis of new algorithms within the generalized hill climbing algorithm framework for address intractable discrete optimization problems. The major technical accomplishments achieved include

i) the introduction of the simultaneous generalized hill climbing (SGHC) algorithm framework for addressing sets of related intractable discrete optimization problems,

ii) the formulation of sufficient conditions under which SGHC algorithms converge,

iii) the formulation of finite-time performance measure for generalized hill climbing (GHC) algorithms.

iv) the application of such measures to a static simulated annealing algorithm, which demonstrated how the performance of such algorithms can be predicted.

v) the application of GHC algorithms to a construction site-leveling problem.

Several other problems were studied during the grant period, including estimation procedures for discrete event simulation problems, aviation security system design and optimization issues, and a pediatric vaccine formulary design problem. All the accomplishments are documented in several archival journal articles and conference proceedings. In addition, many of the results have been presented at national and international conferences, and have won awards for their contribution.

Three Ph.D. dissertations were completed during the period of the grant. Colonel (Sel.) Darrall Henderson successfully defended and submitted his Ph.D. dissertation "Assessing the Finite-Time Performance of Local Search Algorithms" in April 2001. Dr. Tevfik Aytemiz successfully defended and submitted his Ph.D. dissertation " A Probabilistic Study of 3-SATISFIABILITY" in July 2001. Dr. Derek Armstrong successfully defended and submitted his Ph.D. dissertation "A Local Search Algorithm Approach to Analyzing the Complexity of Discrete Optimization Problems" in May 2002.

1. Generalized Hill Climbing Algorithms

Generalized hill climbing algorithms (Jacobson et al. 1998) provide a well-defined framework to model algorithms for intractable discrete optimization problems. Generalized hill climbing algorithms allow inferior solutions to be visited enroute to optimal solutions. This hill climbing capability is the basis for the search strategy's name. To formally describe the generalized hill climbing algorithm framework, several definitions are needed.

For a discrete optimization problem, define the *solution space*, Ω, as a finite set of all possible solutions. Define an *objective function* $f: \Omega \rightarrow [0,+\infty]$ that assigns a non-negative value to each element of the solution space. An important component of GHC algorithms is the *neighborhood function*, $\eta: \Omega \rightarrow 2^{\Omega}$, where $\eta(\omega) \subseteq \Omega$ for all $\omega \in \Omega$. Solutions in a neighborhood are generated *uniformly* at each iteration of a GHC algorithm execution if, for all $\omega \in \Omega$, with $\omega' \in \eta(\omega)$,

$$P\{\omega' \text{ is selected as the neighbor of } \omega \text{ at a given iteration of a GHC algorithm}\} = 1 / |\eta(\omega)|.$$

Unless otherwise stated, assume that the neighbors are generated uniformly at each iteration of a GHC algorithm. The GHC algorithm is described in pseudo-code form (Jacobson et al. 1998):

Set the outer loop counter bound K and the inner loop counter bounds N(k), k = 1,2,...,K
Define a set of hill climbing (random) variables $R_k: \Omega \times \Omega \rightarrow \mathcal{R} \cup \{-\infty,+\infty\}$, k = 1,2,...,K
Set the iteration indices i = 0, k = n = 1
Select an initial solution $\omega(0) \in \Omega$
Repeat while $k \leq K$
  Repeat while $n \leq N(k)$
    Generate a solution $\omega \in \eta(\omega(i))$ and calculate $\delta(\omega(i),\omega) = f(\omega) - f(\omega(i))$
    If $R_k(\omega(i),\omega) \geq \delta(\omega(i),\omega)$, then $\omega(i+1) \leftarrow \omega$ (accept improving or hill climbing moves)
    If $R_k(\omega(i),\omega) < \delta(\omega(i),\omega)$, then $\omega(i+1) \leftarrow \omega(i)$ (reject hill climbing moves)
    $n \leftarrow n+1$, $i \leftarrow i+1$
  Until n = N(k)
  $n \leftarrow 1$, $k \leftarrow k+1$
Until k = K

Note that the outer and inner loop bounds, K and N(k), k = 1,2,...,K, respectively, may all be fixed, or K can be fixed and the N(k), k = 1,2,...,K, are random variables with values determined by the solution at the end of each set of inner loop iterations satisfying some property (e.g., the solution is a local optima). Moreover, assume that the hill climbing random variables have finite means and finite variances for all k and for all possible pairs of elements in the solution space (i.e., $E[R_k(\omega(i),\omega)] < +\infty$ and $Var[R_k(\omega(i),\omega)] < +\infty$ for all $\omega(i) \in \Omega$, $\omega \in \eta(\omega(i))$, and for all k = 1,2,...,K, i = 1,2,...,$I = \sum_{k=1}^{K} N(k)$).

The neighborhood function establishes relationships between the solutions in the solution space, hence allows the solution space to be traversed or searched by moving between solutions. To ensure that the solution space is not fragmented, assume that all the solutions in the solution space (with neighborhood function η) are *reachable;* that is, for all $\omega',\omega'' \in \Omega$, there exists a set of solutions $\omega_1, \omega_2, \ldots, \omega_m \in \Omega$ such that $\omega_r \in \eta(\omega_{r-1})$, r = 1,2,...,m+1, where $\omega' \equiv \omega_0$ and $\omega'' \equiv \omega_{m+1}$. Note that if all solutions in a solution space are reachable, then the solution space (with neighborhood function η) is said to be reachable. The goal is to identify a globally optimal solution ω^* (i.e., $f(\omega^*) \leq f(\omega)$ for all $\omega \in \Omega$).

2. Simultaneous Generalized Hill Climbing Algorithms

One accomplishment during the term of this grant has been the development of a mathematical framework for simultaneously addressing a set of related discrete optimization problems using local search algorithms. The resulting algorithms, termed *simultaneous generalized hill climbing* (SGHC) algorithms (Vaughan and Jacobson 2003a), can be applied to a wide variety of sets of related

manufacturing, military and service industry discrete optimization problems. Many well-known local search algorithms can be embedded within the SGHC algorithm framework, including simulated annealing, threshold accepting, Monte Carlo search, and pure local search (among others).

A SGHC algorithm probabilistically moves between the set of related discrete optimization problems during its execution according to a problem generation probability function. The problem generation probability function is shown to be a stochastic process that satisfies the Markov property. Therefore, given a SGHC algorithm, movement between discrete optimization problems can be modeled as a Markov chain. Sufficient conditions that guarantee that this Markov chain has a uniform stationary probability distribution are obtained. Computational results are presented with SGHC algorithms for a set of traveling salesman problems. For comparison purposes, GHC algorithms are also applied individually to each traveling salesman problem. These computational results suggest that optimal/near optimal solutions can be reached more effectively using a SGHC algorithm.

It is common to encounter several discrete optimization problems where a relationship exists between the solution spaces of the individual problems. In general, these problems are approached individually. However, because of their similarities, the same computational tools can be effectively used to address them simultaneously. For example, the Material Process Design Branch of the Air Force Research Laboratory, Wright Patterson Air Force Base, has several similar discrete manufacturing process design optimization problems under study (see Jacobson et al. 1998, Sullivan and Jacobson 2000). These problems are difficult to solve due in part to the large number of design sequences and associated input parameter setting combinations that exist. Local search algorithms in the *generalized hill climbing* (GHC) algorithm framework were introduced to address such manufacturing design problems (Jacobson et al. 1998). Initial results with GHC algorithms required the manufacturing process design sequence to be fixed, with the GHC algorithm used to identify optimal input parameter settings for each feasible design sequence (Jacobson et al. 1998).

The SGHC algorithm framework is motivated by a study reported in Vaughan et al. (2000) which develops a new neighborhood function that allows a local search algorithm (in the GHC algorithm framework) to be used to also identify the optimal discrete manufacturing process design sequence among a set of valid design sequences. This neighborhood function allows the GHC algorithm to simultaneously optimize over both the design sequences and the input parameters, eliminating the need to approach each design sequence (i.e., a discrete optimization problem) individually. The computational results in Vaughan et al. (2000) suggest that such an approach is feasible and yields reasonable results. However the neighborhood function developed for this purpose is overly complex and problem specific.

The following analysis generalizes the results reported in Vaughan et al. (2000) by formally defining a class of sets of discrete optimization problems where a relationship exists, similar to the one described for the manufacturing problem. A set of discrete optimization problems contained in this class is defined as a set of *fundamentally related* discrete optimization problems. The SGHC algorithm framework is used to simultaneously approach sets of fundamentally related discrete optimization problems using GHC algorithms. A metric between elements in a set of fundamentally related discrete optimization problems is introduced to evaluate if and when it is advantageous to address a particular set of discrete optimization problems with a SGHC algorithm.

Vaughan et al. (2000) introduces a new neighborhood function for simultaneously addressing a set of related manufacturing process design optimization problems using GHC algorithms. This neighborhood function allows for simultaneous optimization across the design sequences and the controllable input parameters. The application of such optimization algorithms (that simultaneously explore multiple manufacturing process designs) using computer simulation is a new advance in how optimal manufacturing process designs can be efficiently identified (Vaughan et al. 2000).

It is common to encounter several discrete optimization problems where a relationship between the solution spaces of the individual problems exists. For example, Henderson, Vaughan and Jacobson (2003) introduces a multiple platform search and rescue optimization problem that is modeled as several discrete optimization problems. They show that for this set, the solution spaces of the individual problems overlap

and are a set of fundamentally related discrete optimization problems. Vaughan et al. (2000) describes an integrated blade rotor discrete manufacturing process design problem that illustrates how certain manufacturing problems can be modeled as several discrete optimization problems with overlapping solution spaces that satisfy the definition of fundamentally related discrete optimization problems. Note that this paper relaxes the methodology used to address the integrated blade rotor discrete manufacturing process design problem described in Vaughan et al. (2000) to develop a general mathematical framework for simultaneously approaching sets of fundamentally related discrete optimization problems.

To discuss the class of sets of discrete optimization problems for which SGHC algorithms are applicable, the following definitions are needed. Consider a *set of discrete optimization problems* $S = \{D_1, D_2, \ldots, D_m\}$, where each discrete optimization problem $D_y = (\Omega_y, f_y)$ is fully defined by a finite set of solutions Ω_y and a real-valued objective function $f_y: \Omega_y \rightarrow R$. A set of discrete optimization problems S is *fundamentally related* by a set $Ob = \{c_1, c_2, \ldots, c_n\}$ of objects if the solution space Ω_y of each discrete optimization problem $D_y = (\Omega_y, f_y) \in S$ can be fully defined by exactly one subset of *Ob*. Then for every discrete optimization problem $D_y = (\Omega_y, f_y) \in S$, there is exactly one set $C_y \subseteq Ob$ such that C_y completely defines Ω_y. The set C_y is defined to be the *fundamental relation set* of D_y.

Let S be a set of fundamentally related discrete optimization problems related by *Ob*. Consider $D_y \in S$ where $C_y \subseteq Ob$ is the fundamental relation set of D_y. Then C_y can be represented by the *binary activity vector* $\mathbf{c}^y \in \{0, 1\}^n$, $\mathbf{c}^y = (B_1, B_2, \ldots, B_n)$, where

$$B_i = \begin{cases} 1, & \text{if } c_i \text{ is contained in } C_y \\ 0, & \text{otherwise} \end{cases}.$$

SGHC algorithms are developed to approach sets of fundamentally related discrete optimization problems. When two discrete optimization problems, D_y and D_q, are contained in a set of fundamentally related discrete optimization problems with respective fundamental relation sets, C_y and C_q, where $|C_y \cap C_q| / |Ob|$ is close to one, it is reasonable to conjecture that the optimal/near optimal solutions of D_y and D_q are similar. The following detachment metric is designed to determine if two discrete optimization problems, in a set of fundamentally related discrete optimization, are close together, hence have similar solution spaces.

Let S be a set of fundamentally related discrete optimization problems related by *Ob*. To formally define the detachment metric ρ between discrete optimization problems $D_y, D_q \in S$, consider the metric space $<\Sigma^n, \rho>$, with $\Sigma = \{0, 1\}$, where the detachment metric ρ is defined on $\Sigma^n \times \Sigma^n$ such that the *distance* between two discrete optimization problems can be determined by considering their binary activity vectors $\mathbf{c}^y = (c_1^y, c_2^y, \ldots, c_n^y) \in \Sigma^n$ and $\mathbf{c}^q = (c_1^q, c_2^q, \ldots, c_n^q) \in \Sigma^n$. Define the *detachment metric* as

$$\rho(D_y, D_q) = \left|c_1^y - c_1^q\right| + \left|c_2^y - c_2^q\right| + \ldots + \left|c_n^y - c_n^q\right|$$

(Royden 1988, p.140). The detachment metric provides a way to measure the overlap (or lack of overlap) between the members in a set of fundamentally related discrete optimization problems.

To apply SGHC algorithms, a neighborhood function with an associated problem generation probability function for moving between discrete optimization problems during an execution of a SGHC algorithm must be developed. The neighborhood function is defined such that each discrete optimization problem has the entire set of discrete optimization problems as neighbors. Therefore, whenever this neighborhood function is applied, every discrete optimization problem is a *candidate problem* (i.e., has a positive probability of being selected). The *problem generation probability function* determines the probability of selecting a candidate problem.

More formally, define the neighborhood function, $\eta_{set}: S \rightarrow 2^S$, such that $\eta_{set}(D_y) = S$, for all $D_y \in S$. Define the *problem generation probability function* $h_{D_y D_q}(k, \rho(D_y, D_q))$, such that

$$0 < h_{D_y D_q}(k, \rho(D_y, D_q)) < 1, \text{ for every } D_y \in S, D_q \in \eta_{set}(D_y),$$

where

$$\sum_{D_q \in \eta_{set}(D_y)} h_{D_y D_q}(k, \rho(D_y, D_q)) = 1, \text{ for every } D_y \in S, D_q \in \eta_{set}(D_y)$$

for every k = 1, 2, …, K.

Note that the problem generation probability function (the probability of selecting a candidate problem, $D_q \in \eta_{set}(D_y)$, $D_y \in S$) can be a function of both the outer loop iteration k = 1,2,…,K and the detachment metric $\rho(D_y, D_q)$.

SGHC algorithms provide a mathematical framework for addressing several fundamentally related discrete optimization problems simultaneously using GHC algorithms. SGHC algorithms seek to find optimal solutions for sets of fundamentally related discrete optimization problems by allowing the algorithm to probabilistically move between discrete optimization problems. When a new discrete optimization problem is generated, an initial solution for this new problem is also generated using information from the previous discrete optimization problem's final solution. The inner and outer loop structure defined for GHC algorithms can be used in SGHC algorithms, where SGHC algorithms restrict possible movement between discrete optimization problems to the first iteration of the outer loop iterations. This constraint ensures that a GHC algorithm is applied to each discrete optimization problem at least N(k) iterations each time it is generated (i.e., initially visited). Note that this was not the case for the manufacturing problem presented in Vaughan et al. (2000), where movement between discrete optimization problems was possible during all inner loop iterations. The SGHC algorithm is presented below in pseudo-code form.

Set the outer loop counter bound K and the inner loop counter bounds N(k), k = 0,1,2,…,K
Define a set of hill climbing (random) variables R_k: $\Omega \times \Omega \rightarrow \Re \cup \{-\infty,+\infty\}$, k = 1,2,…,K
Set the iteration indices N(0) = i = 0, k = n = 1
Select an initial discrete optimization problem D(0) ∈ S and an initial solution ω(0,0) ∈ Ω(0)
Repeat while k ≤ K
    Generate a discrete optimization problem $D(k) \in \eta_{set}(D(k-1))$
    If D(k) ≠ D(k-1), generate a solution ω ∈ Ω(k) and ω(k, 1) ← ω
    Else ω(k, 1) ← ω(k-1, N(k-1))
    Repeat while n ≤ N(k)
        Generate a solution ω ∈ η(ω(k, i))
        Compute δ(ω(k, i),ω) = f(ω)-f(ω(k, i))
        If R_k(ω(k, i),ω) ≥ δ(ω(k, i),ω), then ω(k, i+1) ← ω
        If R_k(ω(k, i),ω) < δ(ω(k, i),ω), then ω(k, i+1) ← ω(k, i)
        n ← n+1, i ← i+1
    Until n = N(k)
    n ← 1, k ← k+1
Until k = K

All SGHC algorithms are formulated using three components, a set of hill climbing random variables $\{R_k\}$, a neighborhood function η between solutions, and a neighborhood function η_{set} between discrete optimization problems. The two-tuple (k,) represents the inner loop iteration i = 1,2,…,N(k), during outer loop iteration k = 1,2,…,K. D(k) is the discrete optimization problem the algorithm is executing over during the k^{th} outer loop iteration, k = 1,2,…,K, where the solution space of D(k) is depicted by Ω(k).

Markov chain theory is an effective tool for studying the performance of local search algorithms. The following analysis shows that an application of the SGHC algorithm can be modeled using Markov chains. In particular, an application of the GHC algorithm is first modeled using a Markov chain. Then an application of the SGHC algorithm is modeled by a set of Markov Chains.

To show that an application of the GHC algorithm can be modeled with a Markov chain, the following definitions are needed. A *stochastic process* is a family of random variables defined on some state space. If there are countable many members of the family, the process (termed a *discrete-time process*) is denoted

by $Q_1, Q_2, \ldots$, where the set of distinct values assumed by a stochastic process is the *state space*. If the state space is countable or finite, the process is a *chain*. A stochastic process $\{Q_k\}$, k = 1,2,... with state space $\Omega = \{\omega_1, \omega_2, \ldots\}$ satisfies the *Markov property* if for every n and for all states $\omega_1, \omega_2, \ldots, \omega_n$

$$\Pr\{Q_n = \omega_n \mid Q_{n-1} = \omega_{n-1}, Q_{n-2} = \omega_{n-2}, \ldots, Q_1 = \omega_1\} = \Pr\{Q_n = \omega_n \mid Q_{n-1} = \omega_{n-1}\} = P_{n(n-1)}.$$

A discrete-time stochastic process that satisfies the Markov property is a *Markov chain* (Isaacson and Madsen 1985).

Let $\{Q_k\}$ denote a discrete-time Markov chain with finite solution space $\Omega = \{\omega_1, \omega_2, \ldots, \omega_{|\Omega|}\}$. For this chain there are $|\Omega|^2$ transition probabilities, $\{P_{ij}\}$, i,j = 1, 2,..., $|\Omega|$. The *transition matrix* associated with the Markov chain $\{Q_k\}$ is P, where P_{ij} is the probability of moving from state ω_i to state ω_j.

An application of a GHC algorithm can be modeled by a stochastic process $\{Q_n^k\}$, k = 1,2,...,K, n = 1,2,...,N(k), $Q_n^k \in \Omega$ with solution space $\Omega = \{\omega_1, \omega_2, \ldots, \omega_{|\Omega|}\}$ that satisfies the Markov property for every n and all states $\omega_1, \omega_2, \ldots, \omega_n$ (i.e., $\{Q_n^k\}$ is a Markov chain). To see this, consider an application of a GHC algorithm to a discrete optimization problem with solution space $\Omega = \{\omega_1, \omega_2, \ldots, \omega_{|\Omega|}\}$. Define $g_{ij}(k)$ to be the *generation probability function* for the neighborhood function η, where $g_{ij}(k)$ is the probability that $\omega_j \in \eta(\omega_i)$ is generated during outer loop iteration k. Consider the inner loop iterations for fixed outer loop iteration k = 1, 2,..., K. Let $\{Q_n^k\}$, k = 1, 2,..., K, n = 1, 2,..., N(k), $Q_n^k \in \Omega$, be the stochastic process where if $Q_n^k = \omega_i$, then the GHC algorithm is at solution ω_i during inner loop iteration n and outer loop iteration k (Johnson and Jacobson 2002a,b). If the GHC algorithm is at solution ω_i at inner loop iteration n-1, the probability that the algorithm is at solution ω_j at inner loop iteration n is

$$P_{ij}(k) = \begin{cases} g_{ij}(k)\Pr(R_k(\omega_i, \omega_j)) \geq \delta_{ij} & \text{for all } \omega_i \in \Omega,\ \omega_j \in \eta(\omega_i),\ j \neq i \\ 1 - \sum_{\substack{z \in \eta(\omega_i) \\ z \neq i}} P_{iz}(k) & j = i \\ 0 & \text{otherwise} \end{cases},$$

independent of the solutions the algorithm visited at inner loop iterations 1,2,...,n-2. Therefore

$$\Pr\{Q_n^k = \omega_j \mid Q_{n-1}^k = \omega_i,\ Q_{n-2}^k = \omega_{i(n-2)}, \ldots,\ Q_1^k = \omega_{i(1)}\} = \Pr\{Q_n^k = \omega_j \mid Q_{n-1}^k = \omega_i\} = P_{ij}(k),$$

Hence the Markov property holds. Moreover, for every outer loop iteration k, the Markov chain $\{Q_n^k\}$ has a transition matrix P(k), where $P_{ij}(k)$ is as defined above .

Recall, that a SGHC algorithm is applied to a set of fundamentally related discrete optimization problems. Movement between discrete optimization problems is only possible at outer loop iterations k=1, 2, ..., K. During the inner loop iterations, the SGHC algorithm is executing over the solution space of the current discrete optimization problem using a GHC algorithm

The following analysis shows that for fixed outer loop iteration k = 1,2,...,K, the stochastic process corresponding to the SGHC algorithm solution at inner loop iterations n = 1,2,...,N(k) can be modeled by a Markov chain that corresponds to an application of a GHC algorithm. Moreover, it is shown that for outer loop iterations k = 1,2,...,K, the possible movement between discrete optimization problems is a stochastic process that satisfies the Markov property.

Consider an application of a SGHC algorithm to a set of fundamentally related discrete optimization problems $S = \{D_1, D_2, \ldots, D_m\}$, where each discrete optimization problem D_y, y = 1,2,...,m is fully defined by a solution space Ω_y and an objective function f_y (i.e., $D_y = (\Omega_y, f_y)$). Consider the inner loop iterations n = 1,2,...,N(k), for fixed outer loop iteration k = 1,2,...,K. Let $\{Q_n^k(D_y)\}$, k = 1,2,...,K, n = 1,2,...,N(k) be

the stochastic process where if $Q_n^k(D_y) = \omega_i$, then the SGHC algorithm is at solution $\omega_{Ii} \in \Omega_y$ at inner loop iteration n of outer loop iteration k.

Note that, for all inner loop iterations n = 1,2,…,N(k) of an outer loop iteration k = 1,2,…,K, the SGHC algorithm is executing over a particular discrete optimization problem from the set of fundamentally related discrete optimization problems $S = \{D_1, D_2, ..., D_m\}$ using a GHC algorithm. It was shown that any application of a GHC algorithm to a discrete optimization problem can be modeled as a Markov chain. Therefore, for fixed outer loop iteration k, the stochastic processes $\{Q_n^k(D_y)\}$, y = 1,2,…,m, with transition matrices P_y, are the Markov chains that correspond to an application of the GHC algorithm to the discrete optimization problems D_y, y = 1,2,…,m for every n = 1,2,…,N(k) and for all states $\omega_1, \omega_2, \ldots, \omega_{|\Omega y|}$.

Movement between discrete optimization problems is a stochastic process that satisfies the Markov property. To see this, define $\{\Psi(k)\}$, $\Psi(k) \in S$, k = 1,2,… to be the stochastic process where if $\Psi(k) = y$, then during outer loop iteration k, for all inner loop iterations n = 1,2,…,N(k) the SGHC algorithm is executing over solutions contained in the solution space of the discrete optimization problem $D_y = (\Omega_y, f_y)$. If the SGHC algorithm is executing over Ω_y at outer loop iteration k-1, then the probability that the SGHC algorithm is executing over Ω_q during outer loop iteration k is

$$T_{yq}(k) = h_{D_y D_q}(k, \rho(D_y, D_q)),$$

independent of the discrete optimization problems the SGHC algorithm visited at outer loop iterations 1,2,…,k-2 and independent of all preceding inner loop iterations. Therefore,

$$\Pr\{\Psi(k) = q \mid \Psi(k-1) = y,\ \Psi(k-2) = y_{k-2}, \ldots, \Psi(1) = y_1\} = \Pr\{\Psi(k) = q \mid \Psi(k-1) = y\} = T_{yq}(k)$$

and the Markov property holds. Moreover, the Markov chain $\{\Psi(k)\}$ has transition matrix T(k), where $T_{yq}(k)$ is as defined above.

Consider an application of the SGHC algorithm to a set of fundamentally related discrete optimization problems, S. This section presents sufficient conditions that guarantee that a SGHC algorithm will (as k approaches $+\infty$) be executing over the solution space of each discrete optimization problem $D_y \in S$ with probability 1/ |S|, where |S| is the cardinality of S. This result implies that, as k approaches $+\infty$, each discrete optimization problem in $S = \{D_1, D_2, \ldots, D_m\}$ is being explored with equal probability.

This section develops sufficient conditions that place restrictions on the selection of the problem generation probability function $h_{D_y D_q}(k, \rho(D_y, D_q))$. A discrete time Markov chain is a *stationary Markov chain* if the probability of going from one state to another state is independent of the iteration at which the transition is being made (Isaacson and Madsen 1985). That is, let $\{X_n\}$ be a stationary Markov chain with state space $S=\{D_1, D_2, \ldots, D_m\}$. Then for all states D_y and D_q, for all k = -(n-1), -(n-2),…, -1, 0, 1, 2, … , $\Pr\{X_n = D_y \mid X_{n-1} = D_q\} = \Pr\{X_{n+k} = D_y \mid X_{n+k-1} = D_q\}$.

The *long run distribution (stationary probability distribution)* of a stationary Markov chain with corresponding transition matrix T is defined by $\pi = [\pi_1 \quad \pi_2 \quad \cdots \quad \pi_m]$, $\pi \geq 0$, for all i = 1,2,…,m, where $\pi = \pi T$ and $\sum_{i=1}^{m} \pi_i = 1$. Equivalently, the long run distribution of a stationary Markov chain is defined by $\pi = [\pi_1 \quad \pi_2 \quad \cdots \quad \pi_m]$, where

$$\pi_j = \lim_{n \to +\infty} T_{ij}^{(n)}.$$

If Markov chain $\{\Psi(k)\}$ is stationary and has a uniform long run distribution, then as k approaches infinity the SGHC algorithm is executing over the solution space of each discrete optimization problem in $S = \{D_1, D_2, \ldots, D_m\}$ with probability 1 / m = 1 / |S|. Theorem 2.2 provides sufficient conditions for the selection of the problem generation probability function $h_{D_y D_q}(k, \rho(D_y, D_q))$ that guarantee that the Markov chain $\{\Psi(k)\}$ has a uniform long run distribution. Therefore, when the sufficient conditions of Theorem

2.2 hold, the SGHC algorithm will (as k approaches $+\infty$) be in discrete optimization problem $D_y \in S$, y = 1,2,…,m with probability 1 / |S|.

To prove Theorem 2.1, the following definitions are needed. A subset, C, of the state space, S, is *closed* if $P_{ij} = 0$ for all $i \in C$, $j \notin C$. A Markov chain is *irreducible* if there exists no nonempty closed set other than S itself. If S has a proper closed subset, it is *reducible*. State ω_i is said to have *period d* if $P_{ii}^{n} = 0$ whenever n is not divisible by d and d is the greatest integer with this property. A state with period one is said to be *aperiodic* (Isaacson and Madsen 1985).

Theorem 2.1: Consider an application of the SGHC algorithm. Define $h_{D_y D_q}(k, \rho(D_y, D_q)) = h_{D_y D_q}(\rho(D_y, D_q)) = h_{D_q D_y}(\rho(D_y, D_q))$, for every k = 1,2,… . Consider the transition matrix T defined by $T_{yq} = h_{D_y D_q}(\rho(D_y, D_q))$. If the transition matrix T is irreducible and aperiodic, then the Markov chain $\{\Psi(k)\}$ has a uniform long run distribution. Moreover, the long run distribution of $\{\Psi(k)\}$ is $\pi = [1/|S|\ 1/|S|\ \ldots\ 1/|S|]$.

Proof: See Vaughan and Jacobson (2003b).

Theorem 2.1 shows that if the problem generation probability function is chosen such that the Markov chain $\{\Psi(k)\}$ is stationary and the associated transition matrix is symmetric, then the Markov chain $\{\Psi(k)\}$ has a uniform stationary distribution. The second set of sufficient conditions, Theorem 2.2, allows for the development of a nonstationary Markov chain. To present Theorem 2.2, a discussion of weak and strong ergodicity, Lemmas 2.1 and Lemma 2.2 are needed.

For finite nonstationary Markov chains, the transition matrices T(k) that contain the probabilities of moving from state D_y to state D_q at outer loop iteration k, are functions of k. To define weak and strong ergodicity of nonstationary Markov chains, several definitions are needed. Define the *one norm* of a vector $\mathbf{f}=(f_1, f_2,\ldots, f_m)$ by $\| \mathbf{f} \| = \sum_{i=1}^{m} | f_i |$. Define the *infinity norm* of matrix T(k) by $\|T(k)\| = \max_i \sum_{j=1}^{m} | T_{ij} |$ (Atkinson 1989). Let T(1), T(2), ... be the transition matrices for a nonstationary Markov chain with starting vector $f^{(0)}$. Define $f^{(j,k)}=f^{(0)}T(j+1)T(j+2) \ldots T(k)$.

A nonstationary Markov chain is *weakly ergodic* if, for all j,

$$\lim_{n\to+\infty} \sup_{f^{(0)}g^{(0)}} \|f^{(j,n)}-g^{(j,n)}\| = 0,$$

where $f^{(0)}$ and $g^{(0)}$ are starting vectors. A nonstationary Markov chain is *strongly ergodic* if there exists a vector $\mathbf{q} = (q_1, q_2, \ldots, q_m)$, with $\| \mathbf{q} \|=1$ and $q_i \geq 0$, for i=1, 2, …, m such that, for all j,

$$\lim_{n\to+\infty} \sup_{f^{(0)}} \|f^{(j,n)}-\mathbf{q}\| = 0,$$

where $f^{(0)}$ is a starting vector (Isaacson and Madsen 1985). The following result from Isaacson and Madsen (1985) is needed in the proof of Lemma 2.2

Lemma 2.1: Let {T(k)} be a sequence of transition matrices corresponding to a nonstationary weakly ergodic Markov chain. If there exists a corresponding sequence of left eigenvectors $\{\pi(k)\}$, for {T(k)}, satisfying

$$\sum_{k=1}^{+\infty} \| \pi(k) - \pi(k+1) \| <^{+} \infty,$$

then the chain is strongly ergodic and for every j,

$$\lim_{n\to+\infty} \sup_{f^{(0)}} \|f^{(j,n)} - \pi\| = 0,$$

where

$$\lim_{k\to+\infty} \pi(k) = \pi.$$

Proof: See Isaacson and Madsen (1985).

Lemma 2.2 shows that if the problem generation probability function is selected such that the corresponding nonstationary Markov chain $\{\Psi(k)\}$, $\Psi(k) \in S$, $k = 1,2,\ldots$ is weakly ergodic and the transition matrices T(k) are symmetric for every k, then the nonstationary Markov chain is strongly ergodic.

Lemma 2.2: Consider an application of the SGHC algorithm. Assume that the nonstationary Markov chain $\{\Psi(k)\}$, $\Psi(k) \in S$, $k = 1,2,\ldots$ is weakly ergodic and that $h_{D_y D_q}(k, \rho(D_y, D_q)) = h_{D_q D_y}(k, \rho(D_q, D_y))$, for every k. Then the nonstationary Markov chain $\{\Psi(k)\}$ is strongly ergodic and for every j,

$$\lim_{n \to +\infty} \sup_{f^{(0)}} \|f^{(j,n)} - \pi\| = 0,$$

where $\pi = [1/|S|\ 1/|S|\ \ldots\ 1/|S|]$.

Proof: See Vaughan and Jacobson (2003b).

When the assumptions in Lemma 2.2 hold, then the SGHC algorithm will (as k approaches $+\infty$) be executing over the solution space of each discrete optimization problem contained in the set of fundamentally related discrete optimization problems with equal probability. Theorem 2.2 shows that this result implies that for every $\varepsilon > 0$, there exists an outer loop iteration such that for all future outer loop iterations, the SGHC algorithm is executing over the solution space of each discrete optimization problem in the set S with probability $1/|S| \pm \varepsilon$.

Theorem 2.2: Consider an application of the SGHC algorithm. Assume that the Markov chain $\{\Psi(k)\}$, $\Psi(k) \in S$, $k = 1,2,\ldots$ is weakly ergodic and that $h_{D_y D_q}(k, \rho(D_y, D_q)) = h_{D_q D_y}(k, \rho(D_q, D_y))$, for every $k = 1,2,\ldots$ and for every $q,y = 1,2,\ldots,m$ (i.e., the corresponding transition matrix is symmetric). Then for every $\varepsilon > 0$, there exists a $k(\varepsilon) \in Z^+$, such that $1/|S| - \varepsilon \le \Pr\{\Psi(k) = y\} \le 1/|S| + \varepsilon$, for all outer loop iterations $k \ge k(\varepsilon)$ and for every $D_y \in S$, $y = 1,2,\ldots,m$.

Proof: See Vaughan and Jacobson (2003b).

Theorem 2.2 establishes that the SGHC algorithm can be designed to ensure that the probability of visiting each problem is uniformly distributed. The following discussion illustrates this result on a set of fundamentally related discrete optimization problems using the traveling salesman problem. The traveling salesman problem (TSP) is a well-known NP-hard discrete optimization problem (Lawler et al. 1985). The TSP is used to illustrate various local search algorithms because it is useful for modeling a variety of real world problems. For example, traditional applications of the TSP include a variety of vehicle routing and scheduling problems. More recently, applications of the TSP have been expanded to include modern applications like the printing of circuit boards, x-ray crystallography, overhauling of gas turbine engines, and the controlling of industrial robots (Johnson and Jacobson 2002a,b).

To formally describe the TSP (Lawler et al. 1985), define a *graph* to be a finite set of vertices, some pairs of which are joined by edges. A *cycle* in a graph is a set of vertices of the graph, which is such that it is possible to move from vertex to vertex, along edges of the graph, so that all vertices are encountered exactly once, finishing at the start. If a cycle contains all the vertices of the graph, it is called a *Hamiltonian cycle (or tour)*. The TSP is defined as follows (Garey and Johnson 1979).

Traveling Salesman Problem (TSP)

Instance: Given a set of n cities $C = \{c_1, c_2, \ldots, c_n\}$ and a distance matrix D that represents the cost of traveling between the cities in the set C.

Question: Find a Hamiltonian cycle $H = (c'_1, c'_2, \ldots, c'_n)$ that minimizes $f(H) = \sum_{j=1}^{n-1} D(c'_j, c'_{j+1}) + D(c'_n, c'_1)$.

An instance of a TSP is a discrete optimization problem, where the solution space is the set of possible all Hamiltonian cycles (with each tour consisting of n cities), $\Omega = \{\omega_1, \omega_2, \ldots, \omega_{(n-1)!/2}\}$. The objective function value for each solution $\omega_i = (c'_1, c'_2, \ldots, c'_n) \in \Omega$ is the sum of the distances the tour depicts, $f(\omega_i) =$

$\sum_{j=1}^{n-1} D(c'_j, c'_{j+1}) + D(c'_n, c'_1)$. The optimal objective function value represents the shortest distance traveled.

The Multiple Traveling Salesman Problem (MTSP) is introduced to illustrate the application of SGHC algorithms. The MTSP is defined as follows.

Multiple Traveling Salesman Problem (MTSP)

Instance: Given a set of n cities $Ob = \{c_1, c_2, \ldots, c_n\}$, a set of m subsets of Ob, $O = \{C_1, C_2, \ldots, C_m\}$, and a distance matrix D that represents the cost of traveling between the cities in the set Ob.

Question: Find a Hamiltonian cycle $H = (c'_1, c'_2, \ldots, c'_{|\Omega_y|})$ where there exists a C_y, $y = 1,2,\ldots,m$ such that $c'_j \in C_y$, for every $j = 1, 2, \ldots, n$, and $f(H) = \sum_{j=1}^{n-1} D(c'_j, c'_{j+1}) + D(c'_n, c'_1)$ is minimized.

Note that each of the sets $C_y \in O$ represents an instance of the TSP, D_y. Since the TSP can be formulated as a discrete optimization problem, then the MTSP problem can be represented by set of discrete optimization problems $S = \{D_1, D_2, \ldots, D_m\}$. The set $S = \{D_1, D_2, \ldots, D_m\}$ is a set of fundamentally related discrete optimization problems. To see this, note that each discrete optimization problem $D_y \in S$, $y = 1,2,\ldots,m$ is fully defined by $C_y \subseteq Ob = \{c_1, c_2, \ldots, c_n\}$. Therefore, C_y is the fundamental relation set of discrete optimization problem D_y.

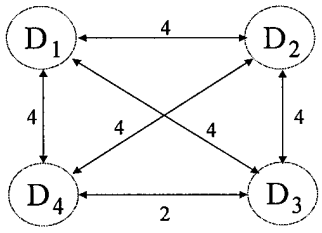
The following computational results illustrate how a MTSP can be addressed with SGHC algorithms. For comparison purposes, GHC algorithms are also applied to the individual members in the set of fundamentally related discrete optimization problems. These computational results suggest that optimal/near optimal solutions can be reached in less total iterations using a SGHC algorithm.

To develop a MTSP, a set consisting of 20 cities was generated by randomly locating each city on a 1000 by 1000 unit grid. Four TSPs of size 18 were generated by randomly selecting 18 of the 20 cities for each traveling salesman problem. In the case where an identical set of cities was generated for two or more of the problems, a completely new set of discrete optimization problems was generated, resulting in four distinct randomly generated TSPs.

The four randomly generated TSPs are arbitrarily labeled D_1, D_2, D_3, and D_4. The detachment metric $\rho(D_y, D_q)$, between the TSPs, was calculated for all $y,q = 1,2,3,4$. The distance matrix and distance diagram are depicted in Figure 2.1.

Figure 2.1: Distance Matrix and Distance Diagram

	D_1	D_2	D_3	D_4
D_1	0	4	4	4
D_2	4	0	4	4
D_3	4	4	0	2
D_4	4	4	2	0



Computational results with Monte Carlo search, pure local search, and simulated annealing using SGHC algorithms are reported. For comparison purposes, computational results with Monte Carlo search, pure local search, and simulated annealing using GHC algorithms are also reported. The 2-Opt neighborhood function was used for all executions of the SGHC and GHC algorithms. For the SGHC algorithms, the problem generation probability function is defined as

$$h_{D_y D_q}(k, \rho(D_y, D_q)) = [1/\rho(D_y, D_q)] / [\sum_{\substack{i=1 \\ i \neq y}}^{4} (1/\rho(D_y, D_i))], \; y \neq q$$

and

$$h_{D_y D_y}(k, \rho(D_y, D_y)) = 1 - \sum_{\substack{q=1 \\ q \neq y}}^{4} [1/(\rho(D_y, D_q)] / [\sum_{\substack{i=1 \\ i \neq y}}^{4} (1/\rho(D_y, D_i))] = 0,$$

for every y,q = 1,2,3,4, $y \neq q$ for every k = 1,2,...,K.

This problem generation probability function is defined such that the associated transition matrix is symmetric and the Markov chain $\{\Psi(k)\}$ is stationary. Therefore, by Theorems 2.1 and 2.2, the Markov chain $\{\Psi(k)\}$ has a uniform stationary distribution, hence as k approaches infinity, the SGHC algorithm is executing over the solution space of each discrete optimization problem in $S = \{D_1, D_2, \ldots, D_m\}$ with probability $1/m = 1/|S| = 1/4$. Moreover, this problem generation function guarantees the discrete optimization problem over which the SGHC algorithm is executing changes at every outer loop iteration k (i.e., $\Psi(k) \neq \Psi(k-1)$, for all k = 1,2,...).

Executions with different values of K and N = N(k), k = 1,2,...,K are reported. To compare the performance of applying a SGHC algorithm versus applying a GHC algorithm, the inner and outer loop bounds of the SGHC algorithm were doubled. Therefore, the total number of iterations that the SGHC algorithm executes is equal to the total number of iterations executed using the GHC algorithm for the four individual problems. Let $R \in Z^+$ represent the total number of replications executed for each SGHC and GHC algorithm formulation. For each replication, a different randomly generated initial tour was used. The means, μ, standard deviations, σ, and the minimum and maximum distances, were computed from the optimal tour distances across these R replications. The value γ in Tables 2.1 to 2.6 represents the number of replications for which the algorithms find the minimum distance tour. For simulated annealing, t_k is updated by multiplying the previous temperature parameter by the increment multiplier $\beta = .90$ (i.e., $t_k = \beta t_{k-1}$), with initial temperature parameter $t_0 = 2000$.

Table 2.1: GHC Algorithm Results: Pure Local Search

Outer and Inner Loop Bounds	γ/R	μ	σ	Minimum	Maximum
K=100, M=100	4/30	3864.3	36.9726	3805.4	3916.0
K=200, M=100	3/30	3863.4	32.7691	3805.4	3907.3
K=300, M=75	1/30	3876.6	36.9549	3805.4	3953.7
K=400, M=75	1/30	3885.3	42.1893	3805.4	3973.4
K=400, M=50	2/30	3867.9	37.7520	3805.4	3916.0
K=800, M=50	1/15	3872.8	35.2469	3805.4	3916.0

Table 2.2: SGHC Algorithm Results: Pure Local Search

Outer and Inner Loop Bounds	γ/R	μ	σ	Minimum	Maximum
K=100, M=100	10/30	3819.4	13.2943	3805.4	3831.8
K=200, M=100	23/30	3808.9	9.0535	3805.4	3831.6
K=300, M=75	23/30	3808.9	9.0535	3805.4	3831.6
K=400, M=75	24/30	3807.2	6.6434	3805.4	3831.6
K=400, M=50	9/30	3818.5	13.3165	3805.4	3831.6
K=800, M=50	12/15	3810.7	10.8417	3805.4	3831.6

The results in Table 2.1 suggest that when the number of outer loop iterations for a pure local search GHC algorithm is increased from 100 to 200, performance of the algorithm shows no improvement. However, Table 2.2 suggests that the performance of a pure local search SGHC algorithm improves significantly (as measured by μ) when the number of outer loop iterations is increased from 100 to 200.

Table 2.3: GHC Algorithm Results: Simulated Annealing

Outer and Inner Loop Bounds	t_0	γ/R	μ	σ	Minimum	Maximum
K=100, M=100	3000	1/30	3854.1	41.4192	3805.4	3953.7
K=200, M=100	3000	1/30	3861.2	35.7662	3805.4	3916.0
K=300, M=75	2000	2/30	3861.5	39.7074	3805.4	3973.4
K=400, M=75	2000	5/30	3855.9	35.3872	3805.4	3907.5
K=400, M=50	2000	3/30	3868.8	39.4863	3805.4	3916.0
K=800, M=50	2000	1/15	3885.9	29.7020	3805.4	3916.0

Table 2.4: SGHC Algorithm Results: Simulated Annealing

Outer and Inner Loop Bounds	T_0	γ/R	μ	σ	Minimum	Maximum
K=100, M=100	3000	16/30	3814.1	12.5549	3805.4	3831.6
K=200, M=100	3000	19/30	3808.0	7.9899	3805.4	3831.6
K=300, M=75	2000	20/30	3808.9	9.0535	3805.4	3831.6
K=400, M=75	2000	23/30	3808.9	9.0535	3805.4	3831.6
K=400, M=50	2000	7/30	3831.6	13.1248	3805.4	3831.6
K=800, M=50	2000	1/15	3813.6	12.5352	3805.4	3831.6

Table 2.5: GHC Algorithm Results: Monte Carlo Search

Outer and Inner Loop Bounds	γ/R	μ	σ	Minimum	Maximum
K=100, M=100	1/30	6390.0	294.6998	5634.9	6805.9
K=200, M=100	1/30	6195.6	239.7325	5575.7	6656.3
K=300, M=75	1/30	6111.7	293.6761	5293.0	6613.9
K=400, M=75	1/30	6099.6	292.5790	5310.2	6488.3
K=400, M=50	1/30	6122.6	287.3693	5291.7	6575.7
K=800, M=50	1/15	6007.1	231.746	5479.1	6328.1

Table 2.6: SGHC Algorithm Results: Monte Carlo Search

Outer and Inner Loop Bounds	γ/R	μ	σ	Minimum	Maximum
K=100, M=100	1/30	6758.2	202.9184	5793.3	6758.2
K=200, M=100	1/30	6109.3	301.5243	5243.7	6611.5
K=300, M=75	1/30	6214.6	204.3092	5727.6	6575.3
K=400, M=75	1/30	6054.3	232.6718	5614.9	6462.8
K=400, M=50	1/30	6265.8	197.6956	5877.3	6659.3
K=800, M=50	1/15	5954.7	301.7151	5299.6	6382.3

The results in Tables 2.5 and 2.6 suggest that there is little difference in performance between Monte Carlo search GHC algorithms and Monte Carlo search SGHC algorithms. Note that this result occurs since Monte Carlo search accepts neighboring solutions independent of their objective function value. Therefore, when movement between discrete optimization problems occurs there is no exchange of common information between the discrete optimization problems. Overall, the results in Tables 2.1 through 2.4 suggest that the SGHC algorithms outperform the GHC algorithms. The minimum distance found over the R replications using SGHC algorithms is significantly smaller than the minimum distance found over the R replications using GHC algorithms for both the simulated annealing and pure local search

algorithms. Additionally, the standard deviation of the optimal values over the R replications is smaller using the SGHC algorithms.

Figures 2.2 through 2.4 depict plots comparing the performance of the SGHC algorithms and the GHC algorithms. To obtain this data, fifteen replications of each SGHC algorithm and GHC algorithm formulation were executed. For each replication, a different randomly generated initial solution was used. The mean of the optimal distances across the fifteen replications for the GHC algorithm are plotted with a solid blue line. The standard deviations of the optimal distances for the GHC algorithm across the fifteen replications are plotted with a dashed blue line. The means of the optimal distances across the fifteen replications for the SGHC algorithm are plotted with a solid red line. The standard deviations of the optimal distances for the SGHC algorithm across the fifteen replications are plotted with a dashed red line. The number of outer loop iterations executed was 800 and the number of inner loop iterations executed was 50 for every formulation. For simulated annealing, t_k is updated by multiplying the previous temperature parameter by the increment multiplier β=.90 with initial temperature parameter t_0=2,000.

Figure 2.2: Pure Local Search

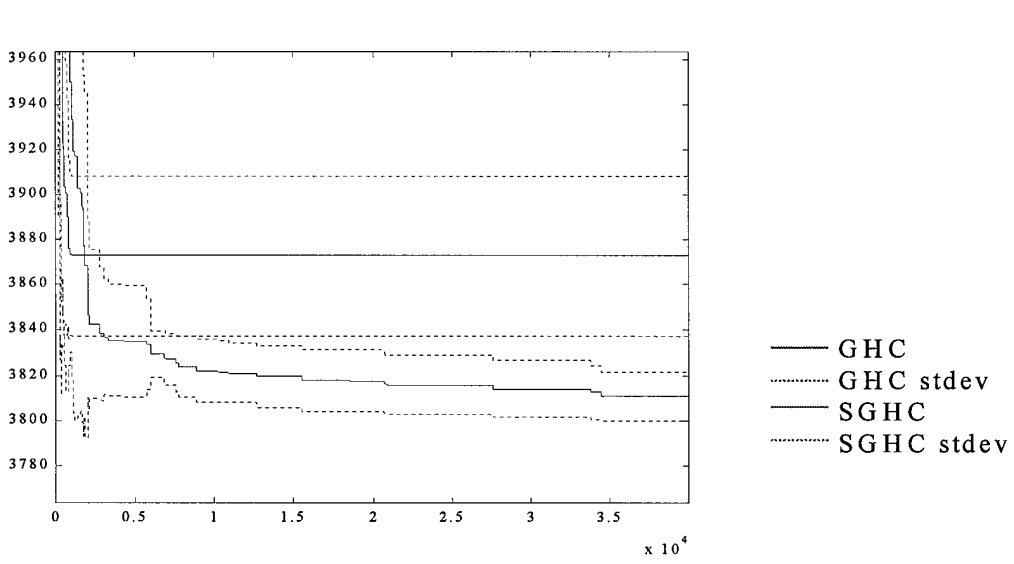


Figure 2.3: Simulated Annealing

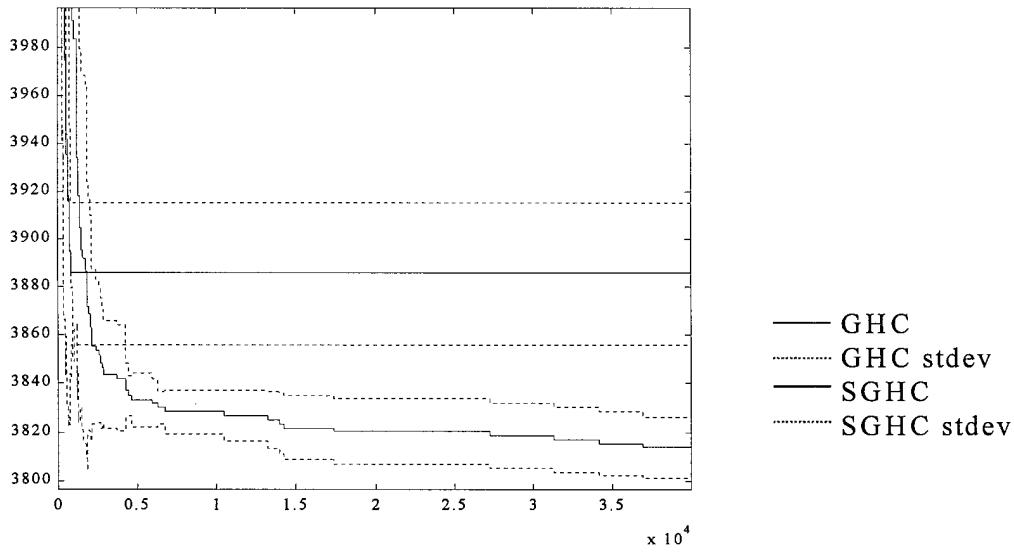
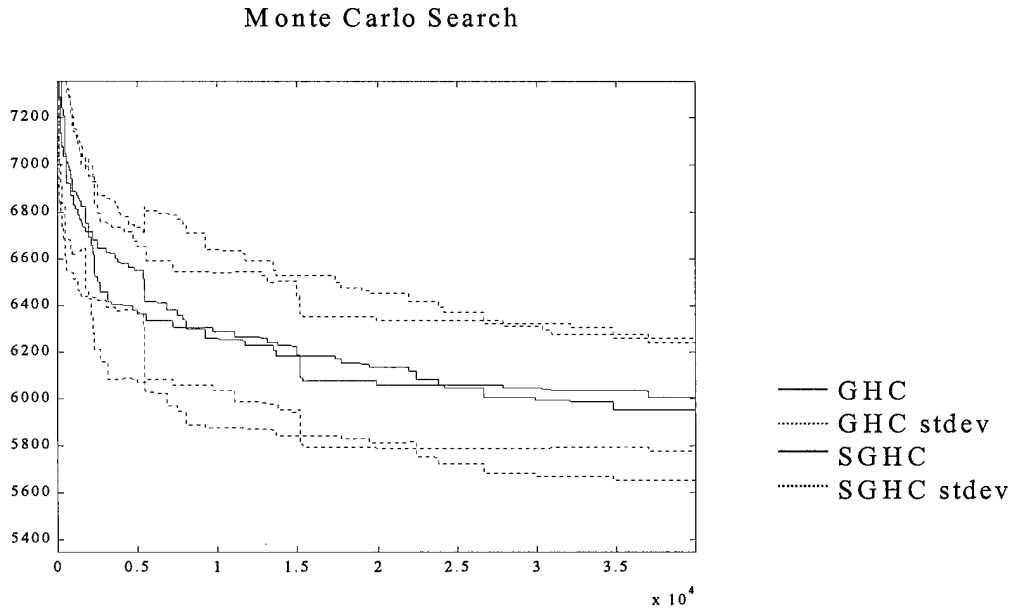


Figure 2.4: Monte Carlo Search



Figures 2.2 and 2.3 suggest that the SGHC algorithms outperform the GHC algorithms, as measured by μ. The minimum distance found over the fifteen replications using SGHC algorithms is significantly smaller after 2500 iterations than the minimum distance found over the fifteen replications using GHC algorithms for both the simulated annealing and pure local search algorithms. Since the SGHC algorithms periodically move between discrete optimization problems they do not get trapped at local minima as frequently as GHC algorithms. Moreover, the standard deviation band of the optimal values over the fifteen replications is smaller using SGHC algorithms. Figure 2.4 suggests that there is no significant difference in the performance of Monte Carlo Search SGHC and GHC algorithms. As noted previously, this result occurs since Monte Carlo search accepts neighboring solutions independent of their objective function value. Therefore, when movement between discrete optimization problems occurs there is no exchange of common information between the discrete optimization problems.

Figures 2.5 through 2.7 depict plots comparing the performance of the SGHC algorithms and the GHC algorithms. To obtain this data, thirty replications of each SGHC algorithm and GHC algorithm formulation were executed. For each replication, a different randomly generated initial solution was used. The mean of the optimal distances across the thirty replications for the GHC algorithm are plotted with a solid blue line. The standard deviations of the optimal distances for the GHC algorithm across the thirty replications are plotted with a dashed blue line. The means of the optimal distances across the thirty replications for the SGHC algorithm are plotted with a solid red line. The standard deviations of the optimal distances for the SGHC algorithm across the thirty replications are plotted with a dashed red line. The number of outer loop iterations executed was 400 and the number of inner loop iterations executed was 75 for every formulation. For simulated annealing, t_k is updated by multiplying the previous temperature parameter by the increment multiplier $\beta = .90$ with initial temperature parameter $t_0 = 2{,}000$.

Figure 2.5: Pure Local Search

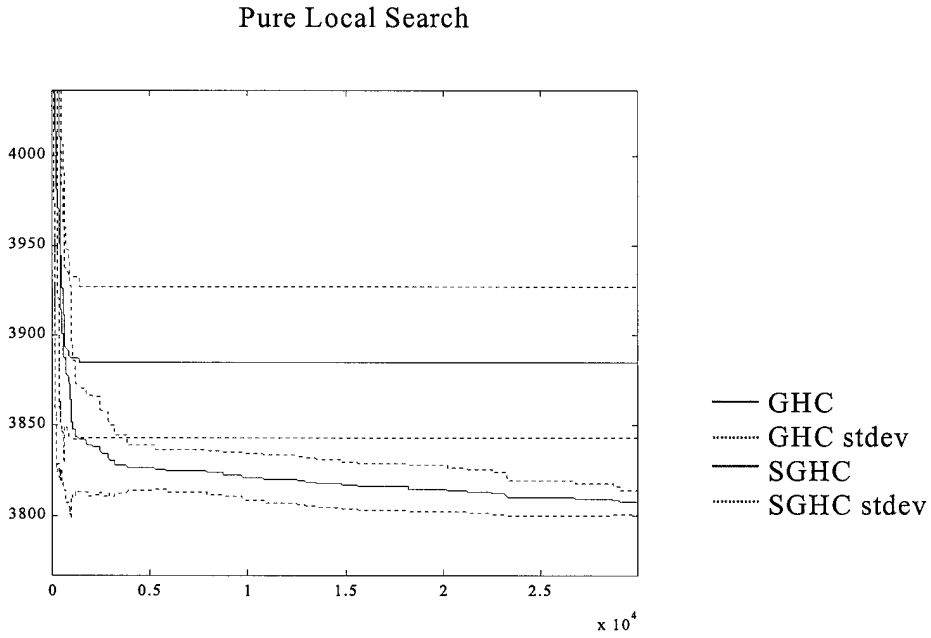


Figure 2.6: Simulated Annealing

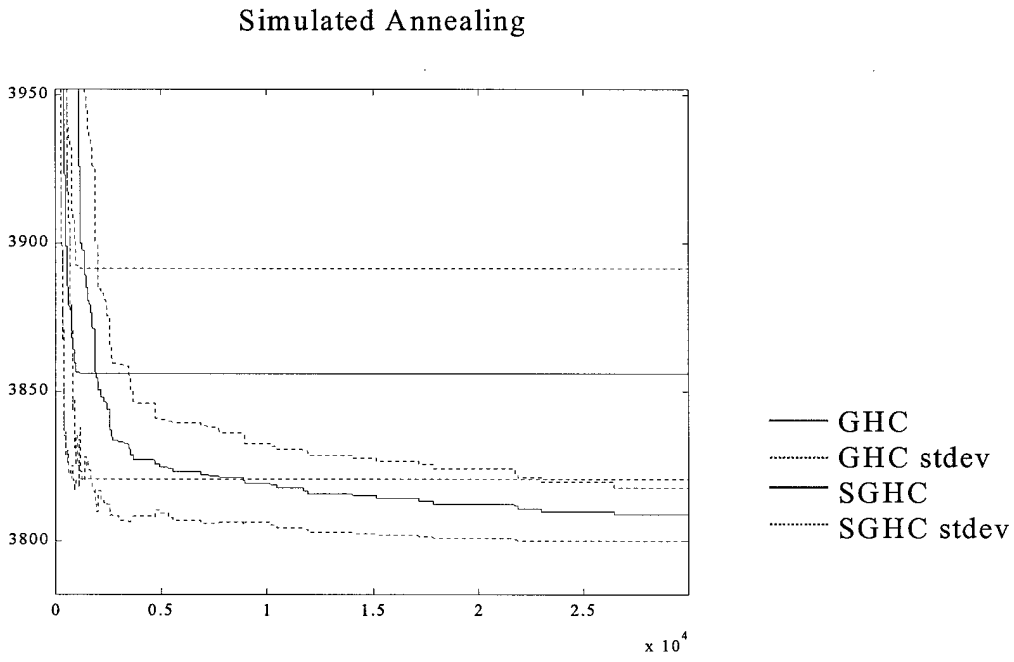
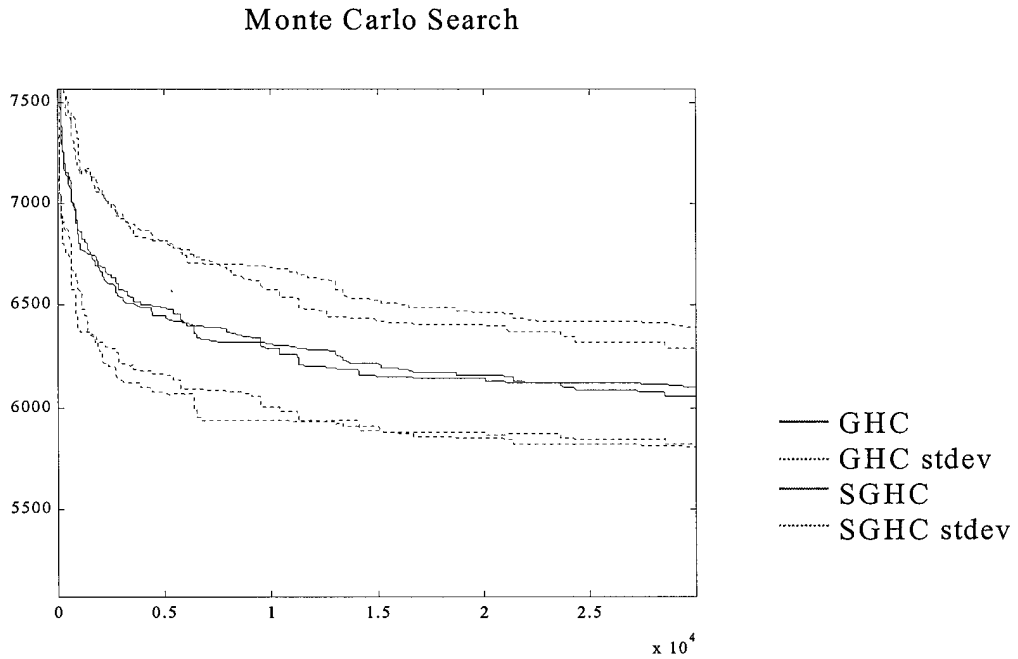


Figure 2.7: Monte Carlo Search



Figures 2.5 and 2.6 suggest that the SGHC algorithms outperform the GHC algorithms, as measured by μ. The minimum distance found over the thirty replications using SGHC algorithms is significantly smaller after 2500 iterations than the minimum distance found over the thirty replications using GHC algorithms for both the simulated annealing and pure local search algorithms. Since the SGHC algorithms periodically move between discrete optimization problems they do not get trapped at local minima as frequently as GHC algorithms. Moreover, the standard deviation band of the optimal values over the thirty replications is smaller using SGHC algorithms. Figure 2.7 suggests that there is no significant difference in the performance of Monte Carlo search SGHC and GHC algorithms. As before, this result occurs since Monte Carlo search accepts neighboring solutions independent of their objective function value. Therefore, when movement between discrete optimization problems occurs there is no exchange of common information between the discrete optimization problems.

The SGHC algorithm is a new approach for addressing a set of fundamentally related discrete optimization problems that can be more efficient than the traditional approach of addressing each discrete optimization problem in the set S individually with a local search algorithm. For example, SGHC algorithms allow practitioners to make a single algorithm run over a set of fundamentally related discrete optimization problems. Moreover, the computational results presented suggest that a SGHC algorithm can outperform the GHC algorithm. The development of the SGHC algorithm and the mathematical results in this paper make it possible for the SGHC algorithm to be adapted and used to approach a variety of real-world problems.

3. β-Acceptable Solution Performance Measures

An important breakthrough has been the introduction and development of β-acceptable solution analysis for GHC algorithms. To obtain these new results, several important research development milestones had to be attained. All these results are reported in Orosz and Jacobson (2002a,b).

The current literature on asymptotic convergence properties and finite-time performance measures focuses primarily on reaching a global minimum. However, in practice, solutions that are *close enough* to a global minimum are often acceptable. Define solutions that have objective function value no greater than β as β-acceptable solutions, where β denotes the maximum acceptable objective function value

(necessarily larger than the global minimum objective function value). This paper analyzes the finite-time behavior of GHC algorithms in reaching β-acceptable solutions.

To illustrate how the β-acceptable solution framework can be applied, a variation of the SA algorithm that uses a fixed cooling schedule term *static simulated annealing* (S^2A) is used. Desai (1999) and Cohn and Fielding (1999) present results regarding the finite-time behavior of the S^2A algorithm that suggest the probability of *visiting* a globally optimal solution is maximized by keeping the temperature fixed rather than allowing the temperature to approach zero. Fielding (2000) presents an insightful empirical study that suggests that there is an optimal fixed temperature for S^2A for different problems. A disadvantage of maintaining a fixed temperature is that the necessary convergence condition (the cooling schedule must approach zero as the number of iterations approaches infinity, e.g., see Hajek 1988) for SA, is violated. During an execution of S^2A, the algorithm may be in a neighborhood of a global optimum, yet not visit this global optimum since there exists a positive probability of not visiting it (i.e., hill climbing). An advantage of S^2A is that there is a nondecreasing positive probability of escaping a local optimum throughout the algorithm's execution.

This research looks at finite-time performance measures for identifying β-acceptable solutions. In particular, the paper derives expression for the expected number of iterations to visit the set of β-acceptable solutions, and uses these expressions to obtain upper and lower bounds for this expectation that can be estimated using finite length runs. These bound estimators are then computed from empirical data generated from independent replications of (moderately short) S^2A algorithm runs. The resulting estimators and the associated estimation procedure provide information that can be used by practitioners to evaluate how long an S^2A algorithm should be run (as measured by the expected number of iterations) to be able to visit the set of β-acceptable solutions for different values of β. Note that the estimation procedure and the need for multiple replications, as in its current form, suggest that it may not be practical to be used for very large problem instances.

The objective function and the neighborhood function for a discrete optimization problem allows the solution space, Ω, to be decomposed into two mutually exclusive and exhaustive sets:

- the set of globally optimal solutions, $G = \{\omega^* \in \Omega: f(\omega^*) \leq f(\omega) \text{ for all } \omega \in \Omega\}$,
- the set of all other solutions, $G^c = \{\omega \in \Omega: f(\omega^*) < f(\omega), \omega^* \in G\}$ where $G \cup G^c = \Omega$.

In many applications, the goal when addressing a discrete optimization problem is to identify a globally optimal solution $\omega^* \in G$. However, from a practical point of view, solutions that are *close enough* to a globally optimal solution (where close enough is measured in terms of the objective function value) for a discrete optimization problem may be acceptable. To describe such solutions, define the set of β-*acceptable solutions*

$$D_\beta = \{\omega \in \Omega: f(\omega) \leq \beta\}, \beta \in \Re. \quad (1)$$

Note that if $\beta < f(\omega^*)$, $\omega^* \in G$, then $D_\beta = \varnothing$. Moreover, if $\beta \geq max_{\omega \in \Omega} f(\omega)$, then $D_\beta = \Omega$. Lastly, $\lim_{\beta \to f(\omega^*)^+} D_\beta = G$, hence G is the upper (right) limit of D_β as β approaches $f(\omega^*)$ from above.

Simulated annealing (SA) is a particular GHC algorithm, with hill climbing random variable, $R_k(\omega, \omega') = -T(k)*\ln(1-U_k)$, $\omega' \in \eta(\omega)$, where the $\{U_k\}$ are IID U(0,1) random variables and T(k), k=1,2,...,K, are the temperate parameters that define the cooling schedule. Therefore, if $f(\omega) - f(\omega(i)) > 0$, then $\omega(i)$ is accepted as the current solution with probability $e^{-[f(\omega') - f(\omega)]/T(k)}$. The temperature parameters are generally set such that they gradually decrease to zero. This means that as k approaches infinity, the SA algorithm behaves similar to pure local search algorithm, hence it will eventually terminate at either a global or a local optimum.

Static simulated annealing (S^2A) is a particular type of SA algorithm, where the temperature is held fixed (or static) during the entire algorithm execution. If $f(\omega) - f(\omega(i)) > 0$, then $\omega(i)$ is accepted as the current solution with probability $e^{-[f(\omega') - f(\omega)]/T}$, where T = T(k), k = 1,2,...,K is held constant at each

iteration. The S^2A and SA pseudo-code are identical, except that only a single temperature T is required for S^2A throughout the entire algorithm execution.

The *one-step β-acceptable probability* is now introduced and described as a finite-time performance measure for GHC algorithms. Define A to be a GHC algorithm applied to an instance of a discrete optimization problem, where $h = \sum_{k=1}^{K} N(k)$ represents the *total* number of algorithm iterations executed. Assume that for algorithm A, $R_k(\omega(i),\omega) \geq 0$, $\omega(i) \in \Omega$, $\omega \in \eta(\omega(i))$, for all iterations $i = 1,2,\ldots,h$, and for all outer loop iterations $k = 1,2,\ldots,K$. At each iteration h, define the event $D(h,\beta)$ on the probability space $(\Sigma, \Im, P)$, termed the set of *β-acceptable solutions*, as

$$D_A(h,\beta) = D(h,\beta) = \{(\omega(1),\omega(2),\ldots,\omega(h)): \omega(i) \in \Omega, i = 1,2,\ldots,h, f(\omega(i)) \leq \beta \text{ for some } i = 1,2,\ldots,h\}$$
$$= \{(\omega(1),\omega(2),\ldots,\omega(h)): \omega(i) \in \Omega, i = 1,2,\ldots,h, \omega(i) \in D_\beta \text{ for some } i = 1,2,\ldots,h\}, \quad (2)$$

where β is an objective function value threshold. Therefore, the complementary event is

$$D_A^c(h,\beta) = D^c(h,\beta) = \{(\omega(1),\omega(2),\ldots,\omega(h)): \omega(i) \in \Omega, i = 1,2,\ldots,h, f(\omega(i)) > \beta \text{ for all } i = 1,2,\ldots,h\}$$
$$= \{(\omega(1),\omega(2),\ldots,\omega(h)): \omega(i) \in \Omega, i = 1,2,\ldots,h, \omega(i) \notin D_\beta \text{ for all } i = 1,2,\ldots,h\}. \quad (3)$$

The event $D(h,\beta)$ defines sequences of h solutions that result from the execution of algorithm A over h iterations, where one or more solutions have objective function values less than or equal to β. The definition of $D(h,\beta)$ implies that $D(i,\beta) \subseteq D(i+1,\beta)$, for all iterations $i = 1,2,\ldots,h$, hence $\{D(h,\beta)\}$ is a telescoping, non-decreasing sequence of events in h. From (2) and (3), the *one-step β-acceptable probability* is defined as

$$r(j,\beta) = P\{D(j,\beta) \mid D^c(j-1,\beta)\} = P\{(\omega(1),\omega(2),\ldots,\omega(j)): \omega(i) \in \Omega, i = 1,2,\ldots,j, f(\omega(i)) > \beta \text{ for all } i = 1,2,\ldots,j-1, f(\omega(j)) \leq \beta\} / P\{D^c(j-1,\beta)\} \quad (4)$$

The one-step β-acceptable probability at iteration j provides a finite-time performance measure for the effectiveness of a GHC algorithm, namely the ability of the algorithm to visit an element of the solution space with objective function value less than or equal to β at iteration j given that it has not already visited such a solution over the first j–1 iterations.

The one-step β-acceptable probability can be used to obtain a closed form expression for (2). Lemma 3.1 captures this relationship.

Lemma 3.1: Consider a GHC algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then $P\{D(h,\beta)\} = 1 - \prod_{j=1}^{h} [1 - r(j,\beta)]$

Proof: See Orosz and Jacobson (2002a).

The one-step β-acceptable probability can be used to express the expected number of iterations to reach the β-acceptable set for the first time. In particular, define the random variable τ_β to represent the minimum number of iterations needed to reach an element in the set of β-acceptable solutions,

$$\tau_\beta \equiv \min\{j \geq 1 : f(\omega(j)) \leq \beta\}. \quad (5)$$

The relationship between τ_β and the one-step β-acceptable probability is described in Lemma 3.2.

Lemma 3.2: Consider a GHC algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\}=1$. Then

(a) $P\{\tau_\beta > h\} = \prod_{j=1}^{h} [1 - r(j,\beta)] = P\{D^c(h,\beta)\}$

(b) $P\{\tau_\beta < +\infty\} = 1 - \prod_{j=1}^{+\infty} [1 - r(j,\beta)]$

(c) $P\{\tau_\beta = h\} = r(h,\beta) \prod_{j=1}^{h-1} [1 - r(j,\beta)]$.

Proof: See Orosz and Jacobson (2002a).

Theorem 3.1 provides an expression for the conditional expectation of τ_β.

THEOREM 3.1: Consider a GHC algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. At iteration h = 1,2,..., if $P\{\tau_\beta \le h\} > 0$ for some $\beta > f(\omega^*)$, $\omega^* \in G$, then

$$E[\tau_\beta \mid \tau_\beta \le h] = h - \sum_{\tau=1}^{h-1} [1 - \prod_{j=1}^{\tau} [1 - r(j,\beta)]] / [1 - \prod_{j=1}^{h} [1 - r(j,\beta)]]$$

Proof: See Orosz and Jacobson (2002a).

The expression in Theorem 3.1 clearly shows that $E[\tau_\beta \mid \tau_\beta \le h]$ is no greater than h. Theorem 3.2 provides an expression for the conditional variance of τ_β.

Theorem 3.2: Consider a GHC algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. At iteration h = 1,2,..., if $P\{\tau_\beta \le h\} > 0$ for some $\beta > f(\omega^*)$, $\omega^* \in G$, then

$$\mathrm{Var}[\tau_\beta \mid \tau_\beta \le h] = (h+1)^2 - \sum_{\tau=0}^{h} [(2\tau + 1)(1 - \prod_{j=1}^{\tau} [1 - r(j,\beta)]) / [1 - \prod_{j=1}^{h} [1 - r(j,\beta)]]]$$
$$- [h - \sum_{\tau=1}^{h-1} [1 - \prod_{j=1}^{\tau} [1 - r(j,\beta)]] / [1 - \prod_{j=1}^{h} [1 - r(j,\beta)]]]^2$$

Proof: See Orosz and Jacobson (2002a).

Theorem 3.3 provides expressions for the conditional expectation of τ_β given that a larger or smaller β value has been reached.

Theorem 3.3: Consider a GHC algorithm execution with initial solution generated such that $P\{D^c(0,\beta_2)\} = 1$. At iteration h = 1,2,..., assume that $P\{\tau_\beta \le h\} > 0$ for some $\beta_1 > f(\omega^*)$ and $\beta_2 > f(\omega^*)$, $\omega^* \in G$, with $\beta_1 < \beta_2$. Then at iteration h,

(a) $$E[\tau_{\beta 1} \mid \tau_{\beta 2} \le h] = \sum_{\tau=1}^{h} [\tau\, r(\tau,\beta_1) \prod_{j=1}^{\tau-1} [1 - r(j,\beta_1)] / [1 - \prod_{j=1}^{h} [1 - r(j,\beta_2)]]]$$
$$+ \sum_{\tau=h+1}^{+\infty} [\tau\, P\{\tau_{\beta 1} = \tau, \tau_{\beta 2} \le h\} / [1 - \prod_{j=1}^{h} [1 - r(j,\beta_2)]]]$$

(b) $$E[\tau_{\beta 2} \mid \tau_{\beta 1} \le h] = (h+1) - \sum_{\tau=0}^{h} [P\{\tau_{\beta 2} \le \tau, \tau_{\beta 1} \le h\} / [1 - \prod_{j=1}^{h} [1 - r(j,\beta_1)]]]$$

Proof: See Orosz and Jacobson (2002a).

The expression for $E[\tau_{\beta 1} \mid \tau_{\beta 2} \le h]$, $\beta_1 < \beta_2$ in Theorem 3.3 provide measures for assessing the finite-time performance of a GHC algorithm. Note that one difficulty with these expressions is that they contain infinite summations, hence are impractical to compute.

Theorem 3.4 provides expressions for the conditional variance of τ_β given that a larger or smaller β value has been reached.

Theorem 3.4: Consider a GHC algorithm execution with initial solution generated such that $P\{D^c(0, \beta_2)\} = 1$. At iteration h = 1,2,..., assume that $P\{\tau_\beta \le h\} > 0$ for some $\beta_1 > f(\omega^*)$ and $\beta_2 > f(\omega^*)$, $\omega^* \in G$, with $\beta_1 < \beta_2$. Then at iteration h,

(a) $$\mathrm{Var}[\tau_{\beta 1} \mid \tau_{\beta 2} \le h] = \sum_{\tau=1}^{h} [\tau^2\, r(\tau,\beta_1) \prod_{j=1}^{\tau-1} [1 - r(j,\beta_1)] / [1 - \prod_{j=1}^{h} [1 - r(j,\beta_2)]]]$$
$$+ \sum_{\tau=h+1}^{+\infty} [\tau^2\, P\{\tau_{\beta 1} = \tau, \tau_{\beta 2} \le h\} / [1 - \prod_{j=1}^{h} [1 - r(j,\beta_2)]]]$$

$$- [\sum_{\tau=1}^{h} [\tau\, r(\tau,\beta_1) \prod_{j=1}^{\tau-1} [1-r(j,\beta_1)] / [1 - \prod_{j=1}^{h} [1-r(j,\beta_2)]]]$$

$$+ \sum_{\tau=h+1}^{+\infty} [\tau\, P\{\tau_{\beta 1} = \tau, \tau_{\beta 2} \le h\} / [1 - \prod_{j=1}^{h} [1-r(j,\beta_2)]]]]^2$$

(b) $$\mathrm{Var}[\tau_{\beta 2} \mid \tau_{\beta 1} \le h] = (h+1)^2 - \sum_{\tau=0}^{h} [(2\tau+1)\, P\{\tau_{\beta 2} \le \tau, \tau_{\beta 1} \le h\} / [1 - \prod_{j=1}^{h} [1-r(j,\beta_1)]]]$$

$$- [(h+1) - \sum_{\tau=0}^{h} [\, P\{\tau_{\beta 2} \le \tau, \tau_{\beta 1} \le h\} / [1 - \prod_{j=1}^{h} [1-r(j,\beta_1)]]]]^2$$

Proof: See Orosz and Jacobson (2002a).

These results describe how the one-step β-acceptable probability and the random variable τ_β can be used to obtain finite-time performance measures for GHC algorithms. The main difficulty with these measures is that they are difficult to compute or estimate. The following description and results show how these measures can be computed for a specific GHC algorithm, the S^2A algorithm. Similar results for cyclical simulation annealing (CSA) are reported in Orosz and Jacobson (2002b).

To obtain a run length analysis of S^2A, properties of the S^2A algorithms can be exploited to obtain upper and lower bounds for $E[\tau_\beta]$. To obtain these bounds for a GHC algorithm, the total number of iterations executed can be decomposed into a set of cycles, each of length h, where the cycle number of iteration τ is denoted by

$$j_h(\tau) = \lceil \tau / h \rceil, \tag{6}$$

the *ceiling function* for τ / h (i.e., the smallest integer greater than or equal to τ / h). To obtain the cycle in which the algorithm visits any element in the set of β-acceptable solutions for the first time, define the random variable

$$J_h(\tau_\beta) \equiv \min\{j_h(\tau) \ge 1 : j_h(\tau)h \ge \tau_\beta, \tau = 1,2,\ldots\} = \lceil \tau_\beta / h \rceil. \tag{7}$$

Figure 3.1 depicts the relationship between $j_h(\tau)$ and τ, where $j_h(\tau) = j$ for all values of τ between (j-1)h+1 and jh. Therefore, if the set of β-acceptable solutions is visited for the first time between iterations $(j_h(\tau)-1)h+1$ and $j_h(\tau)h$, then $J_h(\tau_\beta) = j_h(\tau)$.

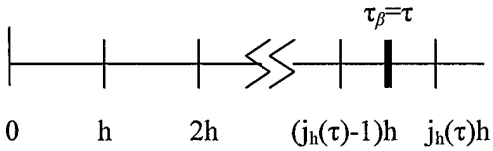


Figure 3.1: Relationship between $j_h(\tau)$ and τ_β

Lemma 3.3 summarizes relationships that exist between the events $\{\tau_\beta = \tau\}$ and $\{J_h(\tau_\beta) = j_h(\tau)\}$.

Lemma 3.3: Consider a GHC algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Suppose that the algorithm is executed for $\tau > 0$ iterations. Then for cycle length $0 < h < \tau$,

(a) $P\{\tau_\beta = \tau\} \le P\{J_h(\tau_\beta) = j_h(\tau)\}$,
(b) $P\{J_h(\tau_\beta) > j_h(\tau)\} = P\{\tau_\beta > j_h(\tau)h\}$,
(c) $P\{J_h(\tau_\beta) > j_h(\tau)\} \le P\{\tau_\beta > j_h(\tau)h - j\}$ for all $j \ge 0$,
(d) $P\{J_h(\tau_\beta) > j_h(\tau)\} \ge P\{\tau_\beta > j_h(\tau)h + j\}$ for all $j \ge 0$,
(e) $P\{J_h(\tau_\beta) = j_h(\tau)\} = P\{\tau_\beta \le j_h(\tau)h\} - P\{\tau_\beta \le (j_h(\tau)-1)h\}$,
(f) $P\{\tau_\beta > j_h(\tau)h\} \le P\{\tau_\beta > \tau\} \le P\{\tau_\beta > (j_h(\tau)-1)h\}$,

Proof: See Orosz and Jacobson (2002a).

Lemma 3.4 obtains an expression for the expectation of $J_h(\tau_\beta)$ for GHC algorithms in terms of the one-step β-acceptable probability.

Lemma 3.4: Consider a GHC algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then

$$E[J_h(\tau_\beta)] = \sum_{j=0}^{+\infty} [\prod_{i=1}^{jh} [1 - r(i,\beta)]].$$

Proof: See Orosz and Jacobson (2002a).

Theorem 3.5 uses the random variable $J_h(\tau_\beta)$ to obtain upper and lower bounds for $E[\tau_\beta]$, the expected number of iterations to visit the set of β-acceptable solutions for the first time.

Theorem 3.5: Consider a GHC algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. Then

$$1 + h\,[E[J_h(\tau_\beta)] - 1] \le E[\tau_\beta] \le 1 + h\,[E[J_h(\tau_\beta)]] \tag{8}$$

Proof: See Orosz and Jacobson (2002a).

Theorem 3.5 provides upper and lower bounds for $E[\tau_\beta]$ that are functions of h and $[E[J_h(\tau_\beta)]$. The problem in applying these bounds is that they contain infinite summations, hence are not easily computable. This problem is difficult to overcome for GHC algorithms in general. However, the cyclical nature of S^2A can be exploited to circumvent this problem. In particular, if the random variable $J_h(\tau_\beta)$ can be modeled as a geometric distribution with parameter $P\{\tau_\beta \le h\}$, then $E[J_h(\tau_\beta)] = 1/P\{\tau_\beta \le h\}$, and the bounds in Theorem 3.5 simplify to

$$1 + h\,[P\{\tau_\beta > h\} / P\{\tau_\beta \le h\}] \le E[\tau_\beta] \le 1 + h\,[1 / P\{\tau_\beta \le h\}].$$

From Lemma 3.2, $P\{\tau_\beta \le h\}$ can be estimated using a finite length execution. Therefore, $E[J_h(\tau_\beta)]$ for S^2A can be estimated using information obtained from replicating each algorithm's performance over a single cycle (i.e., over the first h iterations). To see this, for cycle m, the probability that the algorithm visits the set of β-acceptable solutions for the first time in cycle m is

$$P\{J_h(\tau_\beta) = m\} = P\{(m-1)h + 1 \le \tau_\beta \le mh\}. \tag{9}$$

Lemma 3.5 uses this expression to show that if the one-step β-acceptable probability is constant over all iterations (i.e., $r(i,\beta) = r$ for all i=1,2,...), then $J_h(\tau_\beta)$ is a geometric random variable with parameter $P\{\tau_\beta \le h\} = 1 - (1-r)^h$.

Lemma 3.5: Consider a S^2A algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$ and cycle length $h > 0$. If $r(i,\beta) = r$ for $i = 1,2, \ldots$, then $J_h(\tau_\beta)$ is a geometric random variable with parameter $P\{\tau_\beta \le h\} = 1 - (1-r)^h$.

Proof: See Orosz and Jacobson (2002a).

For S^2A algorithms, since the cooling schedule is fixed at all iterations, then the probability of accepting up hill moves is independent of the iteration number. Therefore, though it is likely to be very difficult to formally prove this result, it may be reasonable to assume that the $r(i,\beta)$ values are constant for all $i = 1,2,\ldots$, hence the result in Lemma 3.5 applies. Theorem 3.6 provides an upper and lower bound on the expected number of iterations to reach the set of β-acceptable solutions, under this assumption.

Theorem 3.6: Consider a S^2A algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. If $J_h(\tau_\beta)$ is a geometric random variable with parameter $P\{\tau_\beta \le h\}$, then

$$1 + h\,[P\{\tau_\beta > h\} / P\{\tau_\beta \le h\}] \le E[\tau_\beta] \le 1 + h / P\{\tau_\beta \le h\}.$$

Proof: See Orosz and Jacobson (2002a).

Theorem 3.7: Consider a S^2A algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\}=1$. If $J_h(\tau_\beta)$ is a geometric random variable with parameter $P\{\tau_\beta \le h\}$, then

$$E[\tau_\beta \mid \tau_\beta \le h]\, P\{\tau_\beta \le h\} + [1 + h / P\{\tau_\beta \le h\}]\, P\{\tau_\beta > h\} \le E[\tau_\beta]$$

$$\leq E[\tau_\beta \mid \tau_\beta \leq h]\, P\{\tau_\beta \leq h\} + [h + 1 + h / P\{\tau_\beta \leq h\}]\, P\{\tau_\beta > h\}$$

and

$$Var[\tau_\beta] \leq h^3([P\{\tau_\beta > h\}+1] / P\{\tau_\beta \leq h\}^2) - (E[\tau_\beta \mid \tau_\beta \leq h]\, P\{\tau_\beta \leq h\} + [1 + h / P\{\tau_\beta \leq h\}]\, P\{\tau_\beta > h\})^2.$$

Proof: See Orosz and Jacobson (2002a).

Theorem 3.8 shows that the upper and lower bounds for $E[\tau_\beta]$ presented in Theorem 3.7 are tighter than the upper and lower bounds for $E[\tau_\beta]$ presented in Theorem 3.6.

Theorem 3.8: Consider a S^2A algorithm execution with initial solution generated such that $P\{D^c(0,\beta)\} = 1$. If $J_h(\tau_\beta)$ is a geometric random variable with parameter $P\{\tau_\beta \leq h\}$, then

$$E[\tau_\beta \mid \tau_\beta \leq h]\, P\{\tau_\beta \leq h\} + [h + 1 + h / P\{\tau_\beta \leq h\}]\, P\{\tau_\beta > h\} \leq 1 + h / P\{\tau_\beta \leq h\}$$

and

$$E[\tau_\beta \mid \tau_\beta \leq h]\, P\{\tau_\beta \leq h\} + [1 + h / P\{\tau_\beta \leq h\}]\, P\{\tau_\beta > h\} \geq 1 + h\,[P\{\tau_\beta > h\} / P\{\tau_\beta \leq h\}].$$

Proof: See Orosz and Jacobson (2002a).

Theorem 3.8 shows that the upper and lower bounds for $E[\tau_\beta]$ in Theorem 3.7 are tighter than the bounds given in Theorem 3.6. Therefore, under the assumption that $J_h(\tau_\beta)$ is a geometric random variable with parameter $P\{\tau_\beta \leq h\}$, the bounds in Theorem 3.7 should be used. Note that under this assumption, these results can only be applied to certain GHC algorithms, such as S^2A. The following description exploits these results for S^2A algorithms to show how the bounds in Theorem 3.7 can be computed.

Computational results with the traveling salesman problem to illustrate how the upper and lower bounds for $E[\tau_\beta]$ in Theorem 3.7 can be computed. The traveling salesman (optimization) problem (TSP) is a well-studied NP-hard discrete optimization problem (Lawler et al. 1985). The diversity of applications for the TSP makes it a frequent choice for testing and evaluating the efficiency and effectiveness of algorithms and heuristics for intractable discrete optimization problems. Traditional applications of the TSP can be found in numerous domains, including vehicle routing and scheduling problems. More recently, it has been applied to the manipulation of robotics (Balaguer et al. 2000), the cutting of industrial components (Foerster and Wascher 1998), and circuit board design (Kobayashi et al. 1999). A search and rescue military application can also be modeled with the traveling salesman problem (Henderson, Vaughan, and Jacobson 2003).

To apply a local search algorithm to an instance of the TSP, a neighborhood function must be defined. There are numerous neighborhood functions that have been devised for the TSP. One such neighborhood function is the *4-change method.* The 4-change method moves between solutions by the exchange of four edges. There are several methods that accomplish the 4-change neighborhood function. One such method is the *city-exchange method.* The city exchange method defines a neighbor of a given cycle by exchanging two cities in the cycle. For example, for a set of seven cities $C = \{c_1, c_2, c_3, c_4, c_5, c_6, c_7\}$ with solution space $\Omega = \{\omega_1, \omega_2, \ldots, \omega_{360}\}$, if the current solution is $\omega = (c'_1, c'_2, c'_3, c'_4, c'_5, c'_6, c'_7)$, then by exchanging cities c'_1 and c'_5, a city exchange neighboring solution is $\omega' = (c'_5, c'_2, c'_3, c'_4, c'_1, c'_6, c'_7)$, with four edges exchanged.

Another commonly used neighborhood function for the TSP is the *2-Opt neighborhood function* (Croes 1958). The 2-Opt neighborhood function moves from one solution to another solution by the exchange of two edges. For example, consider the finite set of seven cities $C = \{c_1, c_2, c_3, c_4, c_5, c_6, c_7\}$, and the corresponding solution space $\Omega = \{\omega_1, \omega_2, \ldots, \omega_{360}\}$. If the current solution is $\omega = (c'_1, c'_2, c'_3, c'_4, c'_5, c'_6, c'_7)$, then one possible neighboring solution is $\omega' = (c'_1, c'_5, c'_4, c'_3, c'_2, c'_6, c'_7)$, which is obtained by reversing the sequence of cities (c'_2, c'_3, c'_4, c'_5). The 2-Opt neighborhood function is a specific version of a more general neighborhood function termed λ-Opt (Helsgaun 2000), where in each move from one solution to another solution, λ edges are exchanged. The λ-opt neighborhood function is based on the concept that a cycle is considered λ-optimal if it is impossible to obtain a lower objective function value by the exchange of any λ edges. From this definition, as λ increases, the resulting local

search algorithm is more likely to find an optimal solution. Unfortunately, the number of operations to test all λ-exchanges also increases as the number of cities increases. Therefore, local search algorithms that use the λ-Opt neighborhood function typically restrict the value for λ to be two or three (Helsgaun 2000).

The Lin-Kernighan neighborhood function (Lin and Kergnighan 1973) operates by determining the λ value in the λ-opt neighborhood function that achieves the best compromise between algorithm run-time and quality of solution. At each iteration of the algorithm, a series of tests is performed to determine the optimal number of λ-exchanges that should be considered. This method is commonly considered the most effective neighborhood function as well as the most difficult to design. Note that since the purpose of the computational experiments is not to determine an optimal neighborhood function for local search algorithms applied to the TSP, but rather, to illustrate the upper and lower bounds derived in Section 4, only the city exchange neighborhood function is used.

The S^2A algorithm was applied to a randomly generated one hundred city instance of the TSP. The distance matrix $D(c_i,c_j)$ between each pair of cities $c_i,c_j \in C$ was generated by randomly placing the cities on a 10×10 Cartesian grid and using a two-dimensional array to store the (X,Y) coordinates of each city (each coordinate was independently generated uniform [0,10]). The distance between cities was then computed and placed in the distance matrix. Note that by design, the distance matrix is symmetric (i.e., $D(c_i,c_j) = D(c_j,c_i)$ for each pair of cities $c_i,c_j \in C$).

The S^2A algorithm was applied with six different (fixed) temperatures (T = 0, 1, 5, 20, 50, 100) and four different cycle lengths (h = 500, 2000, 5000, and 2,000,000), resulting in a total of twenty-four different parameter settings (hence algorithm executions). At each iteration of the S^2A algorithm, a neighbor of the current cycle was generated using the city exchange neighborhood function, where the two cities selected for exchange were uniformly generated across each of the cities (note that experiments using the 2-opt neighborhood function were also performed, but the results obtained were inferior, as measured by the values for $E[\tau_\beta]$, to those obtained using the city exchange neighborhood). For each S^2A algorithm execution, an initial cycle was randomly selected among all possible Hamiltonian cycles. Based on all the S^2A algorithm executions and replications, the smallest Hamiltonian cycle obtained had total distance 99.4, which serves as an upper bound on the optimal Hamiltonian cycle distance. One thousand replications were executed for each of the parameter settings to estimate the one-step β-acceptable probability at each iteration. For each of these replications, the same problem instance and parameter settings were used, with a different randomly generated initial cycle. The average initial solution over the 24,000 runs performed (twenty-four parameter settings with one thousand runs each) had a distance of approximately 479. Each of the twenty-four experiments was executed on a 450 MHz Dell PC. For the cycle lengths h = 500, 2000, 5000, and 2,000,000 the PC completed all six of the fixed temperature settings in approximately 24 seconds, 1.6 minutes, 4 minutes, and 25 hours, respectively. Different β-acceptable values are reported for each parameter setting of T and h. Note that the bounds for only β-acceptable values that are reached over all 1000 replications for the h = 2,000,000 executions are reported; this allows the results for the lower and upper bounds from Theorem 3.7 for smaller values of h to be validated.

Orosz and Jacobson (2002a,b) report extensive computational results with the lower and upper bounds for $E[\tau_\beta]$ in Theorem 3.7 for the S^2A runs with cycle length h = 500, 2000, for fixed temperature values T = 0, 1, 5, 20, 50, and 100. These results suggest that the lower and upper bound estimators are most effective for larger temperatures. Therefore, as the temperature parameter increases, the accuracy for predicting a lower and upper bound on $E[\tau_\beta]$ improves. This is not surprising since as the temperature parameter increases, the S^2A algorithm traverses the solution space more randomly, hence $J_h(\tau_\beta)$ is more likely to be modeled as a geometric random variable with parameter $P\{\tau_\beta \leq h\}$. Furthermore, since the lower and upper bounds and the values for $E[\tau_\beta]$ are estimates, then the upper bound estimates may actually be less than the estimates for $E[\tau_\beta]$. Conversely, the lower bound estimates may actually be greater than the estimates for $E[\tau_\beta]$. This error in the lower and upper bound estimates appears to occur

when $P\{\tau_\beta \leq h\} \leq 0.20$. This may also be due to the geometric distribution assumption on $J_h(\tau_\beta)$ not being valid.

The computational results reported in Orosz and Jacobson (2002a,b) with a one hundred city instance of the TSP validate the lower and upper bound expressions for $E[\tau_\beta]$ in Theorem 3.7. These results suggest that the lower and upper bound expressions for $E[\tau_\beta]$ in Theorem 3.7 provide reasonable measures at higher constant temperature parameter settings. In particular, if similar values of $P\{\tau_\beta \leq h\}$ are compared across different constant temperature parameter settings, T, for the same cycle length, h, then the upper and lower bound estimates for $E[\tau_\beta]$ are most accurate for higher values of T. Moreover, the estimate for $E[\tau_\beta]$ falls between the upper and lower bound estimates for $E[\tau_\beta]$ for approximately 40% of the data reported over all six tables in the upper data set where $P\{\tau_\beta \leq h\} < 1$. However, of the instances where the estimate for $E[\tau_\beta]$ does not fall between the upper and lower bound estimates for $E[\tau_\beta]$, the estimates at *higher* constant temperature parameter settings are much *closer* to the upper and lower bound estimate range than the estimates at *lower* constant temperature parameter settings. These computational results also suggest that the upper and lower bound estimates for $E[\tau_\beta]$ have a small standardized error at higher constant temperature parameter settings, hence provide useful information as estimates for $E[\tau_\beta]$. Lastly, these results suggest that the lower and upper bound expressions for $E[\tau_\beta]$ in Theorem 3.7 provide better measures when estimated with longer cycle lengths. This relationship is reasonable, since as the cycle length increases, the value for $P\{\tau_\beta \leq h\}$ for each value of β also increases.

4. Construction Site Leveling Problem

An interesting application of the research results obtain has been a construction site leveling problem, the details of which are reported in Henderson et al. (2003). To describe the results obtained, the following background information is provided. Heavy engineering and construction projects often require terrain modifications that involve moving large amounts of earth from one area to another area of a construction site. Excavating earth from cut (surplus) locations and hauling it for deposit into fill (deficit) locations requires earthmoving vehicles that are expensive to both operate and maintain, hence often constitute a large portion of a construction project's budget (see Bartholomew 2000). Ideally, planners must develop a strategy for contouring terrain that minimizes the total distance traveled by earthmoving vehicles between cut and fill locations. Minimizing the total distance traveled results in an overall savings to the construction project.

The Shortest Route Cut and Fill Problem (SRCFP) seeks to find a route (beginning and ending at the same cut location) for a single earthmoving vehicle that minimizes the total distance traveled between cut and fill locations. Complete enumeration of all possible solutions to the SRCFP would take a prohibitive amount of time. Therefore it is necessary to construct efficient and effective optimization algorithms to identify optimal/near-optimal routes for construction project planners.

One approach to finding solutions to the SRCFP is to apply a greedy algorithm. A greedy algorithm applies a myopic strategy that begins at an arbitrary cut location, moves to the nearest fill location, and then moves from that fill location to the nearest cut location. This process of moving to the nearest available location continues until the construction project site is leveled (i.e., all the fill locations have been filled with earth from the cut locations). This report summarizes results obtained using this greedy algorithm. The cost associated with the solution found using the greedy algorithm provides an upper bound for the optimal solution for the problem. Local search algorithms have also be used to find near-optimal solutions to the SRCFP. This report also applies simulated annealing (one such local search algorithm) to the SRCFP. The report is organized as follows: First, definitions that are needed to describe the SRCFP and a brief description of the simulated annealing algorithm are presented. The SRCFP is then formally stated as an NP-Hard discrete optimization problem. Computational results for ninety instances of the SRCFP with simulated annealing algorithms are reported. Upper and lower bounds for the optimal objective function value for the SRCFP instances are also given.

To describe the SRCFP, several definitions and terminology are needed. Define a *construction project site* as a plot of land with existing contours that can be modified to facilitate construction of an

object (e.g., road, building, runway). A field survey of the construction project site is typically preformed before any work begins on the construction project. This field survey results in a uniform grid that defines the construction project site by coordinate pairs. Define *locations* on the construction project site as coordinate pairs resulting from a field survey of the construction project site. A *cut location* is a location on the construction project site with excess (surplus) earth and a *fill location* is a location on the construction project site that requires earth (deficit). The *final grade* is achieved when all excess earth at cut locations is removed and all earth deficits at fill locations are filled. Note that not all locations have surplus or deficit earth (i.e., a particular location elevation may be equal to the final grade). Define a *unit load* as the volume of earth that an individual vehicle can carry in one trip (e.g., a sixteen cubic yard scraper versus a thirty cubic yard dump truck). Define the number of *unit cuts* as the number of unit loads available (excess) at a cut location and the number of *unit fills* as the number of unit loads required (deficit) at a fill location. The construction project site contains m unit cuts and n unit fills. To guarantee that a feasible solution exists for the SRCFP, assume that adequate offsite earth is available to ensure that the total deficit can be satisfied and/or that a disposal site is available. This assumption implies $m = n$, since any excess cut (fill) location unit loads can be matched with dummy fill (cut) location unit loads. A *route* is the path (beginning and ending at the same cut location) that an earthmoving vehicle follows to visit every unit cut and every unit fill location exactly once, alternating between the cut and fill locations, until the terrain modifications are complete then returning to initial location. The *total haul distance* is the length of a given route.

The objective of the SRCFP is to find a route that minimizes the total haul distance traveled by a vehicle (i.e., a Hamiltonian circuit with alternating unit cut and unit fill locations that transforms a given piece of terrain into the final grade). Since the SRCFP seeks a route that minimizes the total haul distance while visiting every location exactly once, the SRCFP is a special case of the symmetric traveling salesman problem. The SRCFP can also be related to the capacitated traveling salesman problem with pickups and deliveries (CTSPPD; see Anily and Bramel 1999), since it consists of two location types and uses a vehicle with limited capacity. Therefore, SRCFP is a special case of the swapping problem where the vehicle capacity is equal to one (Anily and Hassin 1992). Chalasani and Motwani (1995) presents a 2-approximation algorithm to address a special case of the swapping problem with two product types, which is equivalent to a special case of the SRCFP with vehicle capacity one.

Note that the SRCFP can be formulated as a transportation problem by relaxing the requirement that a route must alternate between unit cut locations and unit fill locations, and by modifying the objective function to minimize the cost of the distance traveled between locations. The transportation problem optimally allocates resources by minimizing the distance traveled between cut and fill locations; however, it does not take into account the cost of traveling from each fill location to the next cut location. Therefore, relaxing the SRCFP to the transportation problem results in the concept of routes being lost.

Local search algorithms require a neighborhood function at each solution in the solution space. Unfortunately, the very design of local search algorithms means that they often reach a local optimum (for a given neighborhood function), and may not be able to escape this local optimum to continue searching for global optima (e.g., deterministic local search; see Aarts and Lenstra 1997). Simulated annealing (Henderson, Jacobson, and Johnson 2003) is a local search algorithm used to address hard discrete optimization problems. Simulated annealing allows for the escape from local optima, with the possibility of reaching a global optimum, by allowing uphill moves. To describe the implementation of simulated annealing, the following definitions are needed. Let Ω be the solution space (i.e., the set of all possible solutions). Let $f: \Omega \rightarrow \Re$ be the objective function defined on the solution space. The goal is to find a global minima, ω^*, (i.e., $\omega^* \in \Omega$ such that $f(\omega) \geq f(\omega^*)$ for all $\omega \in \Omega$). Note that the objective function is assumed to be bounded to ensure that ω^* exists. Define $\eta(\omega)$ to be the neighborhood function for $\omega \in \Omega$. Therefore, associated with every solution, $\omega \in \Omega$, are neighboring solutions, $\eta(\omega)$, that can be reached in a single iteration of the simulated annealing algorithm. Ω. Define $g_{ij}(k)$ to be the generation probability function for the neighborhood function η, where the probability that $\omega_j \in \eta(\omega_i)$ is generated

during iteration k, is $g_{ij}(k)$. Simulated annealing starts with an initial solution $\omega \in \Omega$. A neighboring solution $\omega' \in \eta(\omega)$ is then generated. If $f(\omega') > f(\omega)$, then ω' is accepted as the current solution with probability $e^{-[f(\omega') - f(\omega)]/T(t)}$, were T(t) is a temperature parameter that is typically non-increasing at each iteration, t.

The SRCFP can be formulated as a discrete optimization problem with the objective of minimizing the total haul distance traveled by an earthmoving vehicle between cut and fill locations. The number of visits to a location depends on the number of unit loads necessary to level the location. Since each unit of earth represents a unit load, the number of visits to each location is the amount of excess (or deficit) earth at that location.

To model the SRCFP as a discrete optimization problem, several definitions are needed. Define $G = \{g_1, g_2, \ldots, g_n\}$ to be a set of n locations. Define $V = \{v_1, v_2, \ldots, v_n\}$ to be a set of volumes for each location in G, (i.e., the number of unit cuts or unit fills). Define $L = \{l_1, l_2, \ldots, l_k\}$ to be the set of single unit cut and unit fill locations, that correspond to locations in G, where $g_i \in G$ appears in L exactly $|v_i|$ times. Therefore, each element in L represents a single (either positive or negative) volume of cut or fill, hence $\sum_{i=1}^{n} |v_i| = k$. Define $L^+ = \{l^+_1, l^+_2, \ldots, l^+_m\}$ to be the set of unit cut locations and $L^- = \{l^-_1, l^-_2, \ldots, l^-_m\}$ to be the set of unit fill locations, where $L \equiv L^+ \cup L^-$, $|L| = k$, and $|L^+| = |L^-| = k / 2 = m$. Recall that the number of unit cut locations equals the number of unit fill locations, and that a unit cut and a unit fill location cannot occur at the same location in G. Define $c_i \in L^+$ to be a single unit cut location, i = 1,2,...,m, and $f_i \in L^-$ to be a single unit fill location, i = 1,2,...,m. Lastly, let $R = (c_1, f_1, \ldots, c_m, f_m)$ represent a route, defined as a Hamiltonian circuit over L with the added constraint that the route must alternate between unit cut locations and unit fill locations. The SRCFP discrete optimization problem is now formally stated.

Shortest Route Cut and Fill Problem (SRCFP)

INSTANCE: Given a set of m unit cut locations, $L^+ = \{l^+_1, l^+_2, \ldots, l^+_m\}$, and a set of m unit fill locations, $L^- = \{l^-_1, l^-_2, \ldots, l^-_m\}$, and a symmetric matrix D containing the distance between the cut and fill locations.

QUESTION: Find a *route* $R = (c_1, f_1, \ldots, c_m, f_m)$, $c_i \in L^+$, i = 1,2,...,m, $f_i \in L^-$, i = 1,2,...,m, such that $g(R) = \sum_{i=1}^{m-1} [D(c_i, f_i) + D(f_i, c_{i+1})] + D(c_m, f_m) + D(f_m, c_1)$ is minimized.

The SRCFP includes the symmetric traveling salesman (optimization) problem (STSP) (Aarts and Lenstra 1997) as a special case. Therefore, the SRCFP is NP-Hard (Garey and Johnson 1979). Define Ω to be the solution space of routes with alternating unit cut and unit fill locations (i.e., $\Omega = \{(c_1, f_1, \ldots, c_m, f_m), c_i \in L^+, f_i \in L^-, i = 1,2,\ldots,m\}$). The cardinality of Ω depends on the number of unit cut and unit fill locations for the given problem instance. Lemma 4.1 provides a closed form expression for this number, and shows that the size of the solution space grows exponentially as the number of unit cut locations and units fill locations increase.

Lemma 4.1: Given an instance of the SRCFP with m unit cut and m unit fill locations, the size of the solution space is $|\Omega| = (m-1)!\, m! / 2$.

Proof: See Henderson et al. (2003).

Computational results were reported with simulated annealing applied to ninety randomly generated instances of the SRCFP. Each instance of the SRCFP corresponds to a construction project site consisting of a square grid with 25, 49, or 81 locations (i.e., a construction site survey resulting in 5 × 5, 7 × 7, and 9 × 9 locations). Thirty instances with each such grid size are reported. The distance matrices are constructed with one unit of distance between each location in the horizontal direction and one unit of distance between each location in the vertical direction. The Euclidean distance between locations i and j, $i, j = 1, 2, \ldots$m, is denoted by D_{ij}. For each instance, the volume of earth (i.e., number of unit cuts or unit fills) at every location is generated as a discrete uniform [-3, 3] random variable. Moreover, each

problem instance includes an offsite location one unit of distance vertically from location l_{11}. This location guarantees the assumption that adequate offsite earth is available to ensure that the total deficit can be satisfied and/or that a disposal site is also available, hence a balanced construction project site is generated (i.e., $m = n$). Note that the random volume of earth generated at each location determines the actual size of the problem instance (i.e., the number of unit cut and unit fill locations). For example, the largest 9 × 9 locations SRCFP problem instance can in theory include as many as 243 total unit cut and unit fill locations, though on average, such problem instances will contain approximately 121.5 unit cut and unit fill locations. From Table 4.3, all thirty randomly generated instances of the SRCFP for this size contained between 100 and 134 total unit cut and unit fill locations, hence provide large SRCFP problem instances to assess the effectiveness of simulated annealing.

For simulated annealing, the cooling schedule {T(t)} is updated by multiplying the previous temperature parameter by an increment multiplier, β, where $0 \leq \beta \leq 1$ (i.e., $T(t) = \beta T(t-1)$). The initial temperature, $T(0) = .2\ (C + F)M$ where C = number of unit cut locations, F = number of unit fill locations and $M = \text{Max}\{D(l_i,l_j),\ i,j = 1, 2,\ldots, m\}$ with $\beta = .98$ for the ninety randomly generated instances of the SRCFP. The neighborhood function η implemented for all experiments is defined as follows: For all routes $R = (c_1, f_1, c_2, f_2, \ldots c_m, f_m)$, the neighbors of R, $\eta(R)$, are defined by selecting any $c', c'' \in \{c_1, c_2, \ldots, c_m\}$, $c' \neq c''$, and reversing the sequence of locations (both cut and fill) between them. Therefore

$$\eta(R) = \{R' \in \Omega : R' = (c_1, f_1, c_2, f_2, \ldots c_{i-1}, f_{i-1}, c_j, f_{j-1}, c_{j-1}, \ldots, f_{i+1}, c_{i+1}, f_i, c_i, f_j, c_{j+1}, f_{j+1}, \ldots, c_m, f_m),$$
$$\text{for some } i,j = 1,2,\ldots,m,\ i < j\}.$$

This neighborhood function is similar to 2-opt for the traveling salesman problem (Aarts and Lenstra 1997). Note that since the distance matrix is symmetric, this neighborhood function includes all routes where two fill locations are selected and the sequence of locations between them are reversed. To generate a neighbor at each iteration, c' is generated uniformly over the set $\{l^+_1, l^+_2, \ldots, l^+_m\}$ and c'' is generated uniformly over the set $\{\{l^+_1, l^+_2, \ldots, l^+_m\} \setminus \{c'\}\}$.

Computational results with simulated annealing are reported for the ninety randomly generated instances of the SRCFP. Simulated annealing was executed with $\tau = 500$ and $N(t) = 2(C+F)$ for $t = 1, 2, \ldots, \tau$. Fifty replications applying simulated annealing to each problem instance were executed. The initial solution for each replication was randomly generated by considering two sets: one of unit cut locations and one of unit fill locations. These two sets are randomly permuted to generate the resulting unit cut and unit fill locations for the initial route. The initial route, $\overline{R}$, was then reconstructed by merging the permuted unit cut and unit fill sets alternating unit cut and unit fill locations. Initial solutions for each replication were obtained in the same manner by randomly permuting the first replication's initial solution.

The mean (μ), the standard deviation (σ), and the minimum and maximum of the objective function values over the fifty replications are reported. Tables 4.1-4.3 report results for the fifty replications of the simulated annealing algorithm for each of the ninety randomly generated instances of the SRCFP.

Table 4.1 Simulated Annealing Results (5 Unit Cut Locations and 5 Unit Fill Locations)					
Problem	(*C*+*F*)	μ	σ	Minimum	Maximum
5x5_1	36	41.36	0.48	41.14	42.37
5x5_2	38	47.74	0.32	47.37	48.02
5x5_3	36	61.04	0.08	60.97	61.40
5x5_4	38	49.16	0.12	49.00	49.25
5x5_5	36	59.57	0.62	58.80	60.73
5x5_6	46	68.36	0.59	68.02	70.13
5x5_7	30	45.14	0.60	44.50	46.50
5x5_8	38	64.48	0.11	64.47	65.27
5x5_9	40	77.24	0.00	77.24	77.24
5x5_10	44	49.08	0.34	48.55	49.96
5x5_11	38	54.72	0.19	54.65	55.24
5x5_12	32	43.71	0.16	43.66	44.25
5x5_13	40	51.50	0.68	51.02	52.44
5x5_14	36	60.88	0.22	60.78	61.57
5x5_15	40	62.56	0.32	62.39	63.19
5x5_16	44	54.14	0.46	53.90	55.31
5x5_17	34	54.27	0.30	54.14	54.90
5x5_18	34	46.38	0.23	46.34	47.51
5x5_19	24	38.54	0.46	38.33	39.51
5x5_20	26	40.60	0.00	40.60	40.60
5x5_21	30	37.14	0.41	36.96	38.31
5x5_22	34	50.26	0.24	49.98	50.68
5x5_23	44	58.66	0.40	58.19	59.74
5x5_24	36	58.44	0.72	58.09	60.83
5x5_25	42	66.86	0.17	66.84	68.06
5x5_26	42	88.04	0.04	88.01	88.21
5x5_27	38	65.42	0.05	65.36	65.47
5x5_28	32	41.48	0.32	41.38	42.44
5x5_29	42	57.32	0.19	57.16	57.78
5x5_30	48	55.48	0.47	54.61	55.86

Table 4.2 Simulated Annealing Results (7 Unit Cut Locations and 7 Unit Fill Locations)					
Problem	(*C*+*F*)	μ	σ	Minimum	Maximum
7x7_1	60	101.45	0.00	101.45	101.45
7x7_2	66	106.34	0.89	105.11	108.01
7x7_3	74	107.17	0.57	106.31	109.27
7x7_4	68	130.00	0.07	129.90	130.13
7x7_5	66	99.45	0.29	98.97	100.30
7x7_6	72	105.32	0.87	104.42	107.79
7x7_7	78	160.77	0.59	160.00	162.11
7x7_8	78	163.89	0.39	163.21	164.65
7x7_9	68	112.13	0.76	111.62	114.17
7x7_10	80	118.76	0.35	118.01	119.59
7x7_11	80	134.47	0.48	133.86	135.82
7x7_12	76	104.70	0.50	103.95	106.19
7x7_13	62	85.50	1.20	84.07	90.52
7x7_14	70	106.22	1.02	105.76	110.82
7x7_15	72	129.37	0.72	128.403	131.42
7x7_16	64	94.22	0.60	93.96	96.33
7x7_17	66	104.67	0.32	104.09	105.41
7x7_18	66	124.96	0.38	124.68	125.97
7x7_19	80	102.39	0.52	101.83	104.89
7x7_20	56	115.50	0.52	115.23	118.58
7x7_21	68	98.23	0.67	97.33	100.32
7x7_22	68	95.40	0.70	94.97	98.44
7x7_23	68	108.68	0.52	108.25	111.54
7x7_24	66	107.42	0.50	106.83	108.83
7x7_25	72	95.95	0.71	94.86	97.04
7x7_26	84	140.17	0.86	139.54	141.79
7x7_27	86	117.90	0.62	117.18	119.21
7x7_28	76	115.20	1.05	113.90	119.60
7x7_29	70	99.73	0.48	99.24	101.46
7x7_30	68	123.96	0.26	123.57	124.92

Table 4.3 Simulated Annealing Results (9 Unit Cut Locations and 9 Unit Fill Locations)					
Problem	$(C+F)$	μ	σ	Minimum	Maximum
9x9_1	122	245.56	0.68	244.62	246.87
9x9_2	132	173.03	1.73	170.17	176.86
9x9_3	124	203.55	0.70	202.25	205.21
9x9_4	116	178.91	1.62	177.00	185.75
9x9_5	118	215.41	0.70	214.34	217.41
9x9_6	132	212.51	1.12	210.53	215.62
9x9_7	126	204.15	0.98	202.53	206.83
9x9_8	134	187.92	1.00	186.83	190.84
9x9_9	116	193.78	0.98	191.87	195.49
9x9_10	100	224.54	0.57	223.66	225.35
9x9_11	120	219.78	0.99	217.93	223.31
9x9_12	116	243.29	0.47	242.49	244.34
9x9_13	116	172.06	0.96	170.40	174.16
9x9_14	112	169.44	1.03	168.10	172.85
9x9_15	122	217.55	0.84	216.33	219.90
9x9_16	112	156.03	1.21	153.78	160.17
9x9_17	110	244.15	0.71	242.87	245.78
9x9_18	108	164.46	1.27	162.67	167.49
9x9_19	128	173.27	1.07	171.72	176.32
9x9_20	130	259.19	0.74	257.70	261.39
9x9_21	132	201.23	1.19	199.71	204.81
9x9_22	116	165.38	0.57	164.62	167.35
9x9_23	136	198.46	1.20	196.65	202.37
9x9_24	120	259.19	0.56	258.58	261.34
9x9_25	120	178.98	1.21	177.40	183.23
9x9_26	124	182.78	1.61	180.81	186.85
9x9_27	132	199.87	0.99	198.22	204.15
9x9_28	102	163.11	0.64	162.26	164.83
9x9_29	114	216.46	0.84	215.14	218.58
9x9_30	118	211.37	0.62	210.42	213.34

To determine if the solution found using the simulated annealing algorithm was indeed optimal, the following integer programming (IP) model of SRCFP is formulated, based on the Miller-Tucker-Zemlin formulation (Miller et al. 1960) of the traveling salesman problem.

$$\text{Min} \sum_{(i,j)\in A} D_{ij}x_{ij}$$

$$\sum_{j\in L^-} x_{ij} = 1 \qquad i \in L^+$$

$$\sum_{j\in L^+} x_{ij} = 1 \qquad i \in L^-$$

$$\sum_{i\in L^-} x_{ij} = 1 \qquad j \in L^+$$

$$\sum_{i\in L^+} x_{ij} = 1 \qquad j \in L^-$$

$$p_i + 1 - k\left(1 - x_{ij}\right) \le p_j \quad \text{for all } (i,j) \in A, \text{with } j \ne 1$$

$$p_1 = 1$$

$$x_{ij} + x_{ji} \le 1 \qquad \text{for all } (i,j) \in A, \text{with } i < j$$

$$x_{ij} \in \{0,1\} \qquad \text{for all } (i,j) \in A$$

$$p_i \in \{1,\ldots,k\} \qquad i = 1,\ldots,k$$

where $L = \{l_1, l_2, \ldots, l_k\}$ is the set of single unit cut and fill locations, L^+ is the set of unit cut locations, L^- is the set of unit fill locations, k is the number of unit cut and fill locations, A is the set of arcs from unit cut locations to unit fill locations and from unit fill locations to unit cut locations, p_i is the position of l_i in the Hamiltonian circuit, and x_{ij} is one if l_i immediately precedes l_j in the Hamiltonian circuit, otherwise x_{ij} is zero.

ILOG OPL Studio 3.0 was used to generate the integer programs for the test problems and CPLEX 6.6 was used to attempt to solve the resulting IP's. The CPLEX parameters were set at their default values, with the time limit set at five hours and the upper cutoff parameter set equal to the best value found by the simulated annealing algorithm plus 0.001. The problems were run on a Pentium III 550 MHZ processor with 128Mb of random access memory.

CPLEX can be used to solve the randomly generated 5x5 problem instances. However, for a randomly generated 7 × 7 problem instance, CPLEX was terminated after four hours of computation because it ran out of random access memory to store the subproblems. During the search, CPLEX explored nearly 34,000 subproblems, but did not find a better solution than the best one found by the simulated annealing algorithm. Furthermore, it appears that the problem cannot be solved in any reasonable length of time even on a faster machine with more memory, since there were approximately 31,000 subproblems waiting to be explored at the time of termination, and this number was still increasing.

Since CPLEX was unable to solve the larger problems, and thus determine if the best solutions found by the simulated annealing algorithms are optimal, the Held-Karp 1-tree lower bound (see Held and Karp 1970, 1971) was computed for each of these problems in order to compute an upper bound on the distance between the best value found by the simulated annealing algorithm and the optimal value. A brief description of this lower bound is included here. Every solution of SRCFP is a Hamiltonian circuit, which can be viewed as a spanning tree of nodes 2, 3,..., k together with two edges incident to node 1. Therefore, the cost of a minimum spanning tree of nodes 2, 3,..., k plus the cost of the two cheapest edges incident to node 1 is a lower bound for the problem. This lower bound can be improved by using Lagrange multipliers on the nodes of the graph. Held et al. (1974) show that subgradient optimization can be used to converge to the optimal set of Lagrange multipliers. Tables 4.4 through 4.6 contain the best value found by the simulated annealing algorithm and the Held-Karp 1-tree lower bounds for the ninety problem instances. They also contain the gap (defined as the difference between the best value found by the simulated annealing algorithm and the lower bound divided by the lower bound). Moreover, a greedy algorithm is presented as an upper bound to assess the effectiveness of simulated

annealing. The greedy algorithm implemented takes an initial unit cut location, c_1, and moves to the nearest (i.e., closest in distance) unvisited unit fill location (i.e., f_j = argmin{$D(c_i, f_j)$, i,j = 1, 2,..., m, f_j has not been visited}. From this unit fill location, f_j, the greedy algorithm moves to the nearest unvisited cut location (i.e., c_i = argmin{$D(f_j, c_i)$, i,j = 1, 2,..., m, c_i has not been visited}. This process continues until the construction project site is leveled, resulting in a paired route of alternating unit cut and unit fill locations. This straightforward technique provides a route that although rarely optimal has an objective function value that is typically much below that of a randomly generated solution.

Table 4.4
Held-Karp 1-Tree Lower Bounds and Greedy Algorithm Results
(5 Unit Cut Locations and 5 Unit Fill Locations)

Problem	SA Best Value	Lower Bound (LB)	Gap between SA Best Value and LB	Greedy Algorithm
5x5_1	41.14	41.136	0.0000	46.54
5x5_2	47.37	47.37	0.0000	54.19
5x5_3	60.97	60.96	0.0002	69.83
5x5_4	49.00	48.74	0.0053	51.36
5x5_5	58.80	58.80	0.0000	75.78
5x5_6	68.02	67.73	0.0043	78.48
5x5_7	44.50	44.41	0.0021	51.94
5x5_8	64.47	64.47	0.0000	72.37
5x5_9	77.24	77.16	0.0009	90.94
5x5_10	48.55	48.25	0.0063	57.14
5x5_11	54.65	54.54	0.0021	73.60
5x5_12	43.66	43.67	0.0000	52.65
5x5_13	51.02	50.81	0.0042	59.84
5x5_14	60.78	60.78	0.0000	71.75
5x5_15	62.39	62.34	0.0008	72.39
5x5_16	53.90	53.53	0.0068	66.72
5x5_17	54.14	54.14	0.0000	65.06
5x5_18	46.34	46.34	0.0000	58.38
5x5_19	38.33	38.07	0.0067	41.75
5x5_20	40.60	40.58	0.0006	44.95
5x5_21	36.96	36.96	0.0000	40.49
5x5_22	49.98	49.97	0.0003	56.17
5x5_23	58.19	58.06	0.0023	63.22
5x5_24	58.09	58.09	0.0000	63.63
5x5_25	66.84	66.83	0.0001	80.03
5x5_26	88.01	87.95	0.0007	95.71
5x5_27	65.36	65.30	0.0010	74.98
5x5_28	41.38	41.34	0.0008	48.43
5x5_29	57.16	57.10	0.0011	65.60
5x5_30	54.61	54.28	0.0062	61.51

Table 4.5 Held-Karp 1-Tree Lower Bounds and Greedy Algorithm Results (7 Unit Cut Locations and 7 Unit Fill Locations)				
Problem	SA Best Value	Lower Bound (LB)	Gap between SA Best Value and LB	Greedy Algorithm
7x7_1	101.45	100.53	0.0092	127.10
7x7_2	105.11	104.80	0.0030	131.37
7x7_3	106.31	105.94	0.0034	136.15
7x7_4	129.90	128.81	0.0085	148.99
7x7_5	98.97	98.63	0.0034	109.57
7x7_6	104.42	103.82	0.0057	128.58
7x7_7	160.00	159.96	0.0002	176.88
7x7_8	163.21	163.19	0.0001	183.90
7x7_9	111.62	111.31	0.0028	124.06
7x7_10	118.01	117.47	0.0046	140.17
7x7_11	133.86	132.61	0.0094	156.50
7x7_12	103.95	103.81	0.0013	124.83
7x7_13	84.07	83.53	0.0064	97.76
7x7_14	105.76	103.97	0.0172	131.18
7x7_15	128.40	127.30	0.0087	169.61
7x7_16	93.96	93.19	0.0082	111.00
7x7_17	104.09	103.53	0.0054	123.38
7x7_18	124.68	124.67	0.0000	151.75
7x7_19	101.83	100.95	0.0087	132.88
7x7_20	115.23	115.23	0.0000	136.16
7x7_21	97.33	96.87	0.0048	104.51
7x7_22	94.97	94.94	0.0003	110.05
7x7_23	108.25	107.84	0.0038	136.03
7x7_24	106.83	106.06	0.0073	118.60
7x7_25	94.86	94.24	0.0066	112.64
7x7_26	139.54	139.29	0.0018	173.34
7x7_27	117.18	115.73	0.0125	133.35
7x7_28	113.90	113.66	0.0021	140.01
7x7_29	99.24	98.55	0.0070	114.49
7x7_30	123.57	123.52	0.0005	147.04

Table 4.6 Held-Karp 1-Tree Lower Bounds and Greedy Algorithm Results (9 Unit Cut Locations and 9 Unit Fill Locations)				
Problem	SA Best Value	Lower Bound (LB)	Gap between SA Best Value and LB	Greedy Algorithm
9x9_1	244.62	243.86	0.0032	311.84
9x9_2	170.17	169.26	0.0054	204.40
9x9_3	202.25	201.04	0.0060	237.06
9x9_4	177.00	176.31	0.0040	222.43
9x9_5	214.34	213.27	0.0050	264.99
9x9_6	210.53	207.80	0.0131	250.71
9x9_7	202.53	200.21	0.0116	245.63
9x9_8	186.83	184.31	0.0137	242.87
9x9_9	191.87	191.62	0.0013	216.29
9x9_10	223.66	222.67	0.0044	267.37
9x9_11	217.93	216.29	0.0076	286.19
9x9_12	242.49	241.16	0.0055	302.43
9x9_13	170.40	168.45	0.0115	217.73
9x9_14	168.10	167.15	0.0057	198.37
9x9_15	216.33	214.90	0.0067	284.97
9x9_16	153.78	151.91	0.0123	204.55
9x9_17	242.87	241.80	0.0044	289.50
9x9_18	162.67	161.78	0.0055	196.99
9x9_19	171.72	170.70	0.0060	194.25
9x9_20	257.70	255.67	0.0079	314.66
9x9_21	199.71	198.26	0.0073	240.30
9x9_22	164.62	161.56	0.0189	203.28
9x9_23	196.65	195.04	0.0083	259.14
9x9_24	258.58	258.29	0.0011	305.55
9x9_25	177.40	175.48	0.0109	228.07
9x9_26	180.81	179.11	0.0095	233.47
9x9_27	198.22	196.28	0.0099	256.40
9x9_28	162.26	162.04	0.0013	197.52
9x9_29	215.14	214.91	0.0010	262.76
9x9_30	210.42	210.20	0.0011	252.72

The best solutions found by the simulated annealing algorithm had an average gap of .18%, .51% and .7% for the randomly generated instances of problems corresponding to grid sizes 5×5, 7×7, and 9×9, respectively. Moreover, all optimal solutions found using simulated annealing were within .68%, 1.72%, and 1.89% of the lower bound for the randomly generated instances of problems corresponding to grid sizes 5×5, 7×7, and 9×9, respectively. Note that the gap is measured relative to a lower bound, not the true optimal solution. It is possible that a significant portion of the gap is due to the lower bound. The best solutions obtained by the simulated annealing algorithm are uniformly better than those obtained by the greedy algorithm.

In the construction industry, using earthmoving vehicles to transform an existing terrain into a final grade (often a level project site) is a costly operation and may represent a significant portion of the overall construction project budget. Reducing the total distance traveled by earthmoving vehicles results in cost savings in terms of fuel consumption, time, and equipment maintenance. This report summarizes the

SRCFP, which seeks to identify a route that minimizes the total haul distance traveled by a vehicle between cut and fill locations and returns the vehicle to its starting location once the final grade is achieved. Finding a route that minimizes the total haul distance is computationally hard. Simplifying the SRCFP to a transportation problem results in the optimal assignment of resources, but does not provide an optimal route. Formulating the SRCFP as a discrete optimization problem and using local search strategies provides near-optimal vehicle routes in a reasonable amount of time.

There are several opportunities to extend the results described in this report. Construction sites and quarries are rarely oriented to allow complete freedom of maneuvering for all vehicle types. Routes are limited to haul roads or vehicle characteristics that make some terrain impassible or prohibitively expensive to traverse, in terms of fuel and/or equipment wear. One assumption in the SRCFP formulation is that the paths between cut and fill location are direct lines (i.e., the equipment is capable of negotiating any terrain without penalty). A more realistic approach is to incorporate haul roads and terrain features into the distance matrix such that a more realistic construction site can be modeled. Work is in progress to collect terrain data from actual construction sites that can be incorporated into future models.

A natural extension of the current problem formulation is to add penalties to routes that force the equipment to negotiate undesirable terrain. Given the choice between avoiding a hill versus going over a hill, the equipment operator's decision is based on cost in terms of fuel, distance or feasibility. Incorporating penalty functions for undesirable terrain may provide more realistic solutions that are of greater interest to the construction industry. Vehicle performance characteristics and fuel consumption rates are available and can also be incorporated into the model. All the modifications described here are meant to add realism to the problem. However, they require extensive data collection that may be difficult to obtain from actual construction projects to support the resulting model.

The same technique used for optimizing cut and fill patterns can be applied to rapid runway repair for both military and nonmilitary airfields. Rapid runway repair is a concern for civilian airports that require maintenance during limited hours of non-use, and with military airfields that require repair from incoming artillery, bombs or missiles. In both cases, time of repair and availability of equipment is the major concern, since busy airports cannot afford to have a runway down for a significant length of time and military runways must be fully mission capable during times of conflict. The SRCFP can be modified to accommodate multiple types of equipment such as earthmoving vehicles as well as concrete and asphalt transporters. Minimizing travel time by developing routes for each type of vehicle between material stockpiles to repair sites is critical to the success of these operations.

Optimal strategies for conducting searches with limited assets are also a natural extension of this model. Different platforms (e.g., helicopters versus fixed wing or satellite) with varying degrees of effectiveness are used for search and rescue operations. Due to the timeliness required in search and rescue operations, efficient use of available assets is critical to success. The optimal routes developed by approaching the SRCFP with local search strategies can be extended to optimize a fleet of search platforms. The same concepts can be extended to optimizing surveillance assets to maximize coverage of a given location.

5. Other Research Results

In addition to the results reported above,, several other results were obtained during the period of the grant. These results are briefly discussed here.

Jacobson, Kobza and Easterling (2001) present a new approach to addressing the difficult tradeoff between false alarms and false clears in aviation security systems. By modeling the problem as a discrete optimization problem, the paper establishes that the resulting problem to be NP-hard, and then develops heuristic procedures to address the problem. The results in this paper have the potential to change the way in which aviation security system information can be synthesized and interpreted. These results may also have important applications in health care (in diagnosing certain types of cancer) and information systems (in detecting when a system has crashed). Note that this paper also received the 2003 Best Paper

award in *IIE Transactions* focused issue on Operations Engineering. Jacobson, Bowman, and Kobza (2001) introduce and analyzes the flight segment baggage value (FSBV) and the passenger segment baggage value (PSBV) performance measures for baggage screening security systems. The paper also includes a real-world example using actual flight data from the Official Airline Guide (OAG) to illustrate an application of the models and results presented in the paper. Virta et al. (2002) look at an important extension of how the origination of selectees can impact the overall security of an air system. These results are particularly noteworthy since two of the 19 hijackers during the 9/11/2001 terrorist activities boarded planes in Portland, Maine and connected to their flights in Boston. The results in this paper identify how such strategies are a major weakness of the air security system, and quantify the risks associated with such security holes. Jacobson et al. (2003) describe how discrete optimization models can be used to address aviation security system deployment and utilization questions, based on three performance measures that quantify the effectiveness of airport baggage screening security device systems. These models are used to solve for optimal airport baggage screening security device deployments considering the number of passengers on a set of flights who have not been cleared using a security risk assessment system in use by the Federal Aviation Administration (i.e., passengers whose baggage is subjected to screening), the number of flights in this set, and the size of the aircraft for such flights. Several examples are provided to illustrate these results, including an example that uses data available from the Official Airline Guide. Virta et al. (2003) consider the cost and benefits of various checked baggage screening strategies. Determining how to effectively operate security devices is as important to overall system performance as developing more sensitive security devices. In light of recent federal mandates for 100% screening of all checked baggage, this paper studies the tradeoffs between screening only selectee checked baggage and screening both selectee and non-selectee checked baggage for a single baggage screening security device deployed at an airport. This tradeoff is represented using a cost model that incorporates the cost of the baggage screening security device, the volume of checked baggage processed through the device, and the outcomes that occur when the device is used. The cost model captures the cost of deploying, maintaining, and operating a single baggage screening security device over a one-year period. The study concludes that as excess baggage screening capacity is used to screen non-selectee checked bags, the expected annual cost increases, the expected annual cost per checked bag screened decreases, and the expected annual cost per expected number of threats detected in the checked bags screened increases. These results indicate that the marginal increase in security per dollar spent is significantly lower when non-selectee checked bags are screened than when only selectee checked bags are screened.

Vazquez-Abad and Jacobson (2001) present a new gradient estimator for the steady-state expected sojourn (system) time in a nonpreemptive priority queueing system. The estimator uses the concept of a phantom system, together with the basic ideas in harmonic gradient estimation, to develop a single simulation run estimator, termed the phantom harmonic gradient estimator. The estimator is shown to be strongly consistent and strongly consistent in the average sense as the sample size grows. An upper bound for the variance of the PHG estimator is presented. This bound is used to show that under mild conditions, the variance of the PHG estimator tends to zero as both the number of phantom systems and the sample size approach infinity. A variance reduction technique that simultaneously uses both common and antithetic random numbers is presented. Computational results on several non-preemptive queueing systems illustrate the effectiveness of the method.

Sullivan and Jacobson (2001) present new necessary and sufficient convergence conditions for generalized hill climbing algorithms. These conditions are then contrasted with similar conditions for simulated annealing algorithms, a particular type of generalized hill climbing algorithms, to show that they reduce to the most often cited and used conditions for such algorithms. These results provide important insights into how the restrictive algorithmic constraints imposed by simulated annealing can be circumvented using relative rates at which global and local optima can be accessed in the solution space by generalized hill climbing algorithms.

Jacobson and Yucesan (2002) identify and investigate common links between discrete-event simulation and discrete optimization algorithms. The primary contribution of the paper is two search problem formulations that are proven to be NP-hard, and which have implication both in the design and analysis of discrete event simulation models and in the design and implementation of discrete optimization algorithms. Fleischer and Jacobson (2002) show how the Boltzmann distribution used in the steady-state analysis of the simulated annealing algorithm gives rise to several scale invariant properties. Scale invariance is first presented in the context of parallel independent processors and then extended to an abstract form based on lumping states together to form new aggregate states. These lumped or aggregate states possess all of the mathematical characteristics, forms and relationships of states (solutions) in the original problem in both first and second moments. These scale invariance properties therefore permit new ways of relating objective function values, conditional expectation values, stationary probabilities, rates of change of stationary probabilities and conditional variances. Such properties therefore provide potential applications in analysis, statistical inference and optimization.

Sewell and Jacobson (2003) is part of an on going research activity into the application of operation research tools in the health care domain. The Recommended Childhood Immunization Schedule has become sufficiently crowded that the prospect of adding additional vaccines to this schedule may not be well received by either health- care providers or parents/guardians. This has encouraged vaccine manufacturers to develop combination vaccines that can permit new vaccines to be added to the schedule without requiring children to be exposed to an unacceptable number of injections during a single clinic visit. This paper develops an integer programming model to assess the economic premium that exists in having combination vaccines available. The results of this study suggest that combination vaccines provide a cost effective alternative to individual vaccines and that further developments and innovations in this area by vaccine manufacturers can provide significant economic and societal benefits. Jacobson, Karnani, and Sewell (2003) report the results of reverse engineering a vaccine selection algorithm to evaluate the economic value of a hepatitis B – *Haemophilus influenzae* type B combination vaccine that is currently under federal contract in the United States. This analysis captures the tradeoff between the cost assigned to administering an injection and the price of the vaccine that earns it a place in the lowest overall cost formulary. Jacobson and Sewell (2002) report other such results using Monte Carlo simulation.

Swisher et al. (2003) present a survey of the literature for two widely used classes of statistical methods for selecting the best design from among a finite set of k alternatives: ranking and selection (R&S) and multiple comparison procedures (MCPs). A comprehensive survey of each topic is presented along with a summary of recent unified R&S-MCP approaches. Procedures are recommended based on their statistical efficiency and ease of application; guidelines for procedure application are offered.

Armstrong and Jacobson (2003) examine the complexity of global verification for MAX-SAT, MAX-k-SAT (for $k \geq 3$), Vertex Cover, and the Traveling Salesman Problem. These results are obtained by adaptations of the transformations that prove such problems to be NP-complete. The class of problems PGS is defined to be those discrete optimization problems for which there exists a polynomial time algorithm such that given any solution ω, either a solution can be found with a better objective function value or it can be concluded that no such solution exists and ω is a global optimum. This paper demonstrates that if any one of MAX-SAT, MAX-k-SAT (for $k \geq 3$), Vertex Cover, or Traveling Salesman Problem are in PGS, then P = NP.

REFERENCES

E.H.L. Aarts, and J.K. Lenstra, 1997, *Local Search in Combinatorial Optimization*, Wiley, New York.

S. Anily, and J. Bramel, 1999, "Approximation Algorithms for the Capacitated Traveling Salesman Problem with Pickups and Deliveries," *Naval Research Logistics*, 46, 654-670.

S. Anily, and R. Hassin, 1992, "The Swapping Problem," *Networks*, 22, 419-433.

D.E. Armstrong, and S.H. Jacobson, 2003, Studying the Complexity of Global Verification for NP-hard Discrete Optimization Problems," *Journal of Global Optimization*, 27(1), 83-96.

K.E. Atkinson, 1989, *An Introduction to Numerical Analysis*, John Wiley and Sons, New York, New York.

C. Balaguer, A. Giménez, J.M. Pastor, V.M. Padrón and M. Abderrahim, 2000, "A Climbing Autonomous Robot for Inspection Applications in 3D Complex Environments," *Robotica*, 18, 287-297.

S. H. Bartholomew, 2000, *Estimating and Bidding for Heavy Construction*, Prentice-Hall, New Jersey.

P. Chalasani, and R. Motwani, 1995, "Approximating Capacitated Routing and Delivery Problems," Working Paper, Department of Computer Science, Stanford University, Palo Alto, CA.

H. Cohn, and M. Fielding, 1999, "Simulated Annealing: Searching for an Optimal Temperature Schedule," *SIAM Journal of Optimization*, 9(3), 779-802.

G.A. Croes, 1958, "A Method for Solving Traveling-Salesman Problems," *Operations Research*, 6, 791-812.

M.P. Desai, 1999, "Some Results Characterizing the Finite Time Behavior of the Simulated Annealing Algorithm," *Sadhana*, 24, 317-337.

M. Fielding, 2000, "Simulated Annealing with an Optimal Fixed Temperature," *SIAM Journal of Optimization*, 11, 289-307.

M.A. Fleischer, and S.H. Jacobson, 2002, "Scale Invariance Properties in the Simulated Annealing Algorithm," *Methodology and Computing In Applied Probability*, 4(3), 219-241.

H. Foerster, and G. Wäscher, 1998, "Simulated Annealing for Order Spread Minimization in Sequencing Cutting Patterns," *European Journal of Operational Research*, 126, 106-130.

M.R. Garey, and D.S. Johnson, 1979, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman and Company, New York, New York.

B. Hajek, 1988, "Cooling Schedules for Optimal Annealing," *Mathematics of Operations Research*, 13, 311-329.

M. Held, and R.M. Karp, 1970, The Traveling Salesman Problem and Minimum Spanning Trees," *Operations Research*, 18, 1138-1162.

M. Held, and R.M. Karp, 1971, "The Traveling Salesman Problem and Minimum Spanning Trees,: Part II," *Mathematical Programming*, 1, 6-25.

M. Held, P. Wolfe and H.P. Crowder, 1974, "Validation of Subgradient Optimization," *Mathematical Programming*, 6, 62-88.

K. Helsgaun, 2000, "An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic," *European Journal of Operational Research*, 126, 106-130.

D. Henderson, S.H. Jacobson, and A.W. Johnson, "The Theory and Practice of Simulated Annealing," 2003, 287-319, (Chapter 10 in *Handbook on Metaheuristics*, F. Glover and G. Kochenberger, Editors), Kluwer Academic Publishers, Norwell, Massachusetts.

D. Henderson, D.E. Vaughan, and S.H. Jacobson, 2003, "Optimal Search Strategies Using Simultaneous Generalized Hill Climbing Algorithms." Technical Report, University of Illinois, Urbana, IL.

D. Henderson, D.E. Vaughan, S.H. Jacobson, R.R. Wakefield, and E.C. Sewell, 2003, "Solving the Shortest Route Cut and Fill Problem using Simulated Annealing," *European Journal of Operational Research*, 145(1), 72-84.

D. Isaacson, and R. Madsen, 1985, *Markov Chains Theory and Applications*, Robert E. Krieger Publishing Company, Inc., Malabar, Florida.

S.H. Jacobson, J.M. Bowman, and J.E. Kobza, 2001, "Modeling and Analyzing the Performance of Aviation Security Systems Using Baggage Value Performance Measures," *IMA Journal of Management Mathematics*, 12(1), 3-22.

S.H. Jacobson, T. Karnani, and E.C. Sewell, 2003, "Analyzing the Economic Value of the Hepatitis B - *Haemophilus Influenzae* Type B Combination Vaccine by Reverse Engineering a Formulary Selection Algorithm," *Vaccine*, 21(17-18), 2178-2186.

S.H. Jacobson, S.H., J.E. Kobza, and A.E. Easterling, 2001, "A Detection Theoretic Approach to Modeling Aviation Security Problems using the Knapsack Problem," *IIE Transactions*, 33(9), 747-759.

S.H. Jacobson, and E.C. Sewell, 2002, "Using Monte Carlo Simulation to Determine Combination Vaccine Price Distributions for Childhood Diseases," *Health Care Management Science*, 5(2), 135-145.

S.H. Jacobson, K.A. Sullivan, and A.W. Johnson, 1998, "Discrete Manufacturing Process Design Optimization Using Computer Simulation and Generalized Hill Climbing Algorithms," *Engineering Optimization*, 31, 247-260.

S.H. Jacobson, J.E. Virta, J.M. Bowman, J.E. Kobza, and J.J. Nestor, 2003, "Modeling Aviation Baggage Screening Security Systems: A Case Study," *IIE Transactions*, 35(3), 259-269.

S.H. Jacobson, and E. Yucesan, 2002, "Common Issues in Discrete-Event Simulation and Discrete Optimization," *IEEE Transactions on Automatic Control*, 47(2), 341-345.

A.W. Johnson, and S.H. Jacobson, 2002a, "On the Convergence of Generalized Hill Climbing Algorithms," *Discrete Applied Mathematics*, 119(1-2), 37-57.

A.W. Johnson, and S.H. Jacobson, 2002b, "A Class of Convergent Generalized Hill Climbing Algorithms," *Applied Mathematics and Computation*, 125(2-3), 359-373.

S. Kobayashi, M. Edahiro, and M. Kubo, 1999, "A VLSI Scan-Chain Optimization Algorithm for Multiple Scan-Paths," *IEICE Transactions on Fundamentals of Electronics, Communications, and Computer Science*, 11, 2499-2504.

E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, 1985, *The Traveling Salesman Problem*, John Wiley and Sons, Chichester.

S. Lin, and B.W. Kernighan, 1973, "An Effective Heuristic for the Traveling Salesman Problem," *Operations Research*, 21, 498-516.

C.E. Miller, A.W. Tucker and R.A. Zemlin, 1960, "Integer Programming Formulations and Traveling Salesman Problems," *Journal of the Association for Computing Machinery*, 7, 326-329.

J.E. Orosz, and S.H. Jacobson, 2002a, "Finite-time Performance Analysis of Static Simulated Annealing Algorithms," *Computational Optimization and Applications*, 21(1), 21-53.

J.E. Orosz, and S.H. Jacobson, 2002b, "Analysis of Static Simulated Annealing Algorithms," *Journal of Optimization Theory and Application*, 115(1), 165-182.

H.L. Royden, 1988, *Real Analysis*, Third Edition, Prentice Hall, New Jersey.

E.C. Sewell and S.H. Jacobson, 2003, "Using an Integer Programming Model to Determine the Price of Combination Vaccines for Childhood Immunization," *Annals of Operations Research*, 119, 261-284.

K.A. Sullivan, and S.H. Jacobson, 2000, "Ordinal Hill Climbing Algorithms for Discrete Manufacturing Process Design Optimization Problems," *Discrete Event Dynamical Systems* 10(4), 307-324.

K.A. Sullivan, and S.H. Jacobson, 2001, "A Convergence Analysis of Generalized Hill Climbing Algorithms" *IEEE Transactions on Automatic Control*, 46(8), 1288-1293.

J.R. Swisher, S.H. Jacobson, and E. Yucesan, 2003, Discrete-event simulation optimization using ranking, selection, and multiple comparison procedures: a survey," *ACM Transactions on Modeling and Computer Simulation*, 13(2), 134-154.

D.E. Vaughan, and S.H. Jacobson, 2003a, "Simultaneous Generalized Hill Climbing Algorithms for Addressing Sets of Discrete Optimization Problems," Technical Report, University of Illinois, Urbana, IL.

D.E. Vaughan, and S.H. Jacobson, 2003b, "Nonstationary Markov Chain Theory for Simultaneous Generalized Hill Climbing Algorithms," Technical Report, University of Illinois, Urbana, IL.

D.E. Vaughan, S.H. Jacobson, and D.E. Armstrong, 2000, "A New Neighborhood Function for Discrete Manufacturing Process Design Optimization Using Generalized Hill Climbing Algorithms," *ASME Journal of Mechanical Design*, 122(2), 164-171.

F.J. Vazquez-Abad, and S.H. Jacobson, 2001, "Phantom Harmonic Gradient Estimators for Priority Queueing Systems," *INFORMS Journal on Computing*, 13(4), 345-359.

J.E. Virta, S.H. Jacobson, J.E. Kobza, 2002, "Outgoing Selectee Rates At Hub Airports," *Reliability Engineering and System Safety,* 76(2), 155-165.

J.E. Virta, S.H. Jacobson, and J.E. Kobza, 2003, "Analyzing the Cost of Screening Selectee and Non-selectee Baggage," *Risk Analysis*, 23(5), 897-908.

CONTRIBUTING PERSONNEL

The principal investigator for this project, Dr. Sheldon H. Jacobson, has devoted 33% of his academic year time, and 33% of his summer, in each of the three years of this project. Dr. Diane Vaughan worked as a post-doctoral research associate on this project. Colonel (Sel.) Darrall Henderson (US Army) has provided input on this project throughout the three years of the grant. Dr. T. Aytemiz, Dr. D. Henderson, Dr. D. Armstrong, Ms. J. Virta, and Mr. J. Orosz were M.S. and Ph.D. students of the principal investigator, who worked on various aspects of this project. Ms. V. Venkat and Ms. L. McLay are current graduate students of the principal investigator who have also contributed to various aspects of this project.

TRANSITIONS

Extensive interactions with Austral Engineering and Software, Incorporated, have resulted in generalized hill climbing algorithm software code that is available for military use.

PUBLICATIONS

The following is a list of published papers in refereed journals or book chapters that are related to the research effort supported in whole or in part by this grant.

Armstrong, D.E., Jacobson, S.H., 2003 "Studying the Complexity of Global Verification for NP-hard Discrete Optimization Problems," *Journal of Global Optimization*, 27(1), 83-96.

Fleischer, M.A., Jacobson, S.H., 2002, "Scale Invariance Properties in the Simulated Annealing Algorithm," *Methodology and Computing In Applied Probability*, 4(3), 219-241.

Henderson, D., Jacobson, S.H., Johnson, A.W., 2003, "The Theory and Practice of Simulated Annealing," (Chapter 10 in *Handbook in Metaheuristics*, F. Glover and G. Kochenberger, Editors), 287-319.

Henderson, D., Vaughan, D.E., Jacobson, S.H., Wakefield, R.R., Sewell, E.C., 2003, "Solving the Shortest Route Cut and Fill Problem Using Simulated Annealing," *European Journal of Operational Research*, 145(1), 72-84.

Jacobson, S.H., Bowman, J.M., Kobza, J.E., 2001, "Modeling and Analyzing the Performance of Aviation Security Systems Using Baggage Value Performance Measures," *IMA Journal of Management Mathematics*, 12(1), 3-22.

Jacobson, S.H., 2002, "Analyzing the Performance of Local Search Algorithms using Generalized Hill Climbing Algorithms," (Chapter 20 in *Essays and Surveys on Metaheuristics*, P. Hansen and C.C. Ribeiro, Editors), Kluwer Academic Publishers, Norwell, Massachusetts, 441-467.

Jacobson, S.H., Karnani, T., Sewell, E.C., 2003, "Analyzing the Economic Value of the Hepatitis B - *Haemophilus Influenzae* Type B Combination Vaccine by Reverse Engineering a Formulary Selection Algorithm, *Vaccine*, 21(17-18), 2178-2186.

Jacobson, S.H., Kobza, J.E., Easterling, A.E., 2001, "A Detection Theoretic Approach to Modeling Aviation Security Problems using the Knapsack Problem," *IIE Transactions*, 33(9), 747-759.

Jacobson, S.H., Sewell, E.C., 2002, "Using Monte Carlo Simulation to Determine Combination Vaccine Price Distributions for Childhood Diseases," *Health Care Management Science*, 5(2), 135-145.

Jacobson, S.H., Virta, J.E., Bowman, J.M., Kobza, J.E., Nestor, J.J., 2003, "Modeling Aviation Baggage Screening Security Systems: A Case Study," *IIE Transactions*, 35(3), 259-269.

Jacobson, S.H., Yucesan, E., 2002, "Common Issues in Discrete-Event Simulation and Discrete Optimization," *IEEE Transactions on Automatic Control*, 47(2), 341-345.

Johnson, A.W., Jacobson, S.H., 2002, "On the Convergence of Generalized Hill Climbing Algorithms," *Discrete Applied Mathematics*, 119(1-2), 37-57.

Johnson, A.W., Jacobson, S.H., 2002, "A Class of Convergent Generalized Hill Climbing Algorithms," *Applied Mathematics and Computation*, 125(2-3), 359-373.

Orosz, J.E., Jacobson, S.H. 2002, "Finite-time Performance Analysis of Static Simulated Annealing Algorithms," *Computational Optimization and Applications*, 21(1), 21-53.

Orosz, J.E., Jacobson, S.H., 2002, "Analysis of Static Simulated Annealing Algorithms," *Journal of Optimization Theory and Application*, 115(1), 165-182.

Sewell, E.C., Jacobson, S.H., 2003, "Using an Integer Programming Model to Determine the Price of Combination Vaccines for Childhood Immunization," *Annals of Operations Research*, 119, 261-284.

Sullivan, K.A., Jacobson, S.H., 2001, "A Convergence Analysis of Generalized Hill Climbing Algorithms" *IEEE Transactions on Automatic Control*, 46(8), 1288-1293.

Swisher, J.R., Jacobson, S.H., Yucesan, E., 2003, "Discrete-event Simulation Optimization Using Ranking, Selection, and Multiple Comparison Procedures: A Survey," *ACM Transactions on Modeling and Computer Simulation*, 13(2), 134-154.

Vazquez-Abad, F.J., Jacobson, S.H., 2001, "Phantom Harmonic Gradient Estimators for Priority Queueing Systems," *INFORMS Journal on Computing*, 13(4), 345-359.

Virta, J.E., Jacobson, S.H., Kobza, J.E., 2002, "Outgoing Selectee Rates At Hub Airports," *Reliability Engineering and System Safety,* 76(2), 155-165.

Virta, J.E., Jacobson, S.H., Kobza, J.E., 2003, "Analyzing the Cost of Screening Selectee and Non-Selectee Baggage," *Risk Analysis*, 23(5), 897-908.

HONORS/AWARDS

Dr. Diane E. Vaughan, a post-doctoral research associate at the University of Illinois, was awarded third place in the 2002 *Institute of Industrial Engineering (IIE) Pritsker Doctoral Dissertation Award* Competition. The purpose of this national competition is to recognize outstanding graduate research in the field of industrial engineering, and thereby promote better industrial engineering research. Dr. Vaughan was also supported as a graduate student on AFOSR grants F49620-98-1-0111 and F49620-98-1-0432.

Dr. Sheldon H. Jacobson was named an *Associate in the Center for Advanced Studies* at the University of Illinois during the 2002-2003 academic year. This competition is open to all tenured faculty at the university. Each academic year, between fifteen and twenty such faculty are awarded this distinction, based on their record of research accomplishments and their plan for future research.

Dr. Sheldon H. Jacobson was named a *Willett Faculty Scholar in the College of Engineering* at the University of Illinois for the 2002-2005 academic years. This designation is given to tenured faculty who are excelling in their contributions to the university. Approximately twenty faculty have been awarded this distinction, based on their record of research accomplishments and achievements.

Dr. Sheldon H. Jacobson was awarded the 2002 *Aviation Security Research Award*, sponsored by Aviation Security International, the International Air Transport Association, and the Airports Council International.

Dr. Sheldon H. Jacobson co-authored the paper that was awarded the 2003 *Best Paper Award* in the *IIE Transactions* Focused Issue on Operations Engineering.

Dr. Sheldon H. Jacobson was awarded a 2003 *Guggenheim Fellowship* by the John Simon Guggenheim Memorial Foundation. A total of 184 awards were made out of a pool of 3282 applications for the 2003 competition. In addition, only two engineers in the entire country were awarded fellowships this year.